S01P0605US00

日 本 国 特 許 庁
PATENT OFFICE
JAPANESE GOVERNMENT

#5
7-9-03

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。
This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日
Date of Application: ２０００年 ４月２１日

出 願 番 号
Application Number: 特願２０００－１２０９４０

出 願 人
Applicant (s): ソニー株式会社

CERTIFIED COPY OF
PRIORITY DOCUMENT

２００１年 ３月 ２日

特 許 庁 長 官
Commissioner,
Patent Office 及 川 耕 造

出証番号 出証特２００１－３０１４２４５

【書類名】　特許願

【整理番号】　0000221906

【提出日】　平成12年 4月21日

【あて先】　特許庁長官　近藤　隆彦　殿

【国際特許分類】　H04L 12/00

H04L 29/00

【発明者】

【住所又は居所】　東京都品川区北品川６丁目７番３５号　ソニー株式会社内

【氏名】　山岸　靖明

【発明者】

【住所又は居所】　東京都品川区北品川６丁目７番３５号　ソニー株式会社内

【氏名】　権野　善久

【発明者】

【住所又は居所】　東京都品川区北品川６丁目７番３５号　ソニー株式会社内

【氏名】　西尾　郁彦

【発明者】

【住所又は居所】　東京都品川区北品川６丁目７番３５号　ソニー株式会社内

【氏名】　角田　智弘

【特許出願人】

【識別番号】　000002185

【氏名又は名称】　ソニー株式会社

【代表者】　出井　伸之

【代理人】

【識別番号】　100082762

【弁理士】

【氏名又は名称】　杉浦　正知

【電話番号】　03-3980-0339

【手数料の表示】

【予納台帳番号】　043812

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【包括委任状番号】　9708843

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　送信装置および送信方法、受信装置および受信方法、ならびに、送受信システムおよび送受信方法

【特許請求の範囲】

【請求項１】　公開鍵証明情報を階層的に管理するディレクトリの階層構造を送信する送信装置において、

自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、上記コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、上記コンテナエントリと上記リーフエントリとからなるディレクトリの階層構造を管理する管理手段と、

上記管理手段によって管理される上記ディレクトリの階層構造の変化を検出し、該検出結果に基づき上記ディレクトリの階層構造の変化の差分からなる差分情報を求める検出手段と、

上記検出手段で検出された上記差分情報を送信する送信手段と

を有し、

上記コンテナエントリおよび／または上記リーフエントリには、最新の公開鍵証明情報を取得可能な情報と、該最新の公開鍵証明情報の失効情報とを格納するようにしたことを特徴とする送信装置。

【請求項２】　請求項１に記載の送信装置において、

上記失効情報は、上記公開鍵証明情報のシリアル番号であることを特徴とする送信装置。

【請求項３】　請求項１に記載の送信装置において、

上記コンテナエントリおよび／または上記リーフエントリの属性に、上記最新の公開鍵証明情報および上記最新の公開鍵証明情報を取得するための情報のうち何れか一方を選択して格納可能としたことを特徴とする送信装置。

【請求項４】　請求項３に記載の送信装置において、

上記検出手段により上記差分が検出された更新事象からの時間に応じて、上記属性に格納される情報を、上記最新の公開鍵証明情報と上記最新の公開鍵証明情報を取得するための情報との間で変更可能なようにしたことを特徴とする送信装

置。

【請求項５】　公開鍵証明情報を階層的に管理するディレクトリの階層構造を送信する送信方法において、

自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、上記コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、上記コンテナエントリと上記リーフエントリとからなるディレクトリの階層構造を管理する管理のステップと、

上記管理のステップによって管理される上記ディレクトリの階層構造の変化を検出し、該検出結果に基づき上記ディレクトリの階層構造の変化の差分からなる差分情報を求める検出のステップと、

上記検出のステップで検出された上記差分情報を送信する送信のステップと

を有し、

上記コンテナエントリおよび／または上記リーフエントリには、最新の公開鍵証明情報を取得可能な情報と、該最新の公開鍵証明情報の失効情報とを格納するようにしたことを特徴とする送信方法。

【請求項６】　送信された、公開鍵証明情報を階層的に管理するディレクトリの階層構造を受信する受信装置において、

送信された、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、上記コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、上記コンテナエントリと上記リーフエントリとからなるディレクトリの階層構造の変化を検出した検出結果に基づき求められた上記ディレクトリの階層構造の変化の差分からなる差分情報を受信する受信手段と、

上記受信手段で受信された上記差分情報に基づき構成されるディレクトリの階層構造を管理する管理手段と、

上記差分情報を選択的に取り込み、上記管理手段で管理される上記ディレクトリの階層構造を変更する変更手段と

を有し、

上記コンテナエントリおよび／または上記リーフエントリには、最新の公開鍵

証明情報を取得可能な情報と、該最新の公開鍵証明情報の失効情報とが格納されて上記送信されることを特徴とする受信装置。

【請求項７】　請求項６に記載の受信装置において、

上記失効情報は、上記公開鍵証明情報のシリアル番号であることを特徴とする受信装置。

【請求項８】　請求項６に記載の受信装置において、

上記変更手段は、上記公開鍵証明情報を取得するための証明情報パスに対応する上記コンテナエントリおよび／または上記リーフエントリの更新情報を上記選択的に取り込むことを特徴とする受信装置。

【請求項９】　送信された、公開鍵証明情報を階層的に管理するディレクトリの階層構造を受信する受信方法において、

送信された、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、上記コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、上記コンテナエントリと上記リーフエントリとからなるディレクトリの階層構造の変化を検出した検出結果に基づき求められた上記ディレクトリの階層構造の変化の差分からなる差分情報を受信する受信のステップと、

上記受信のステップで受信された上記差分情報に基づき構成されるディレクトリの階層構造を管理する管理のステップと、

上記差分情報を選択的に取り込み、上記管理のステップで管理される上記ディレクトリの階層構造を変更する変更のステップと

を有し、

上記コンテナエントリおよび／または上記リーフエントリには、最新の公開鍵証明情報を取得可能な情報と、該最新の公開鍵証明情報の失効情報とが格納されて上記送信されることを特徴とする受信方法。

【請求項１０】　公開鍵証明情報を階層的に管理するディレクトリの階層構造を送信し、送信されたディレクトリの階層構造を受信する送受信システムにおいて、

自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、上

記コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、上記コンテナエントリと上記リーフエントリとからなるディレクトリの階層構造を管理する第１の管理手段と、

上記第１の管理手段によって管理される上記ディレクトリの階層構造の変化を検出し、該検出結果に基づき上記ディレクトリの階層構造の変化の差分からなる差分情報を求める検出手段と、

上記検出手段で検出された上記差分情報を送信する送信手段と、

上記送信手段により送信された、上記差分情報を受信する受信手段と、

上記受信手段により受信された上記差分情報に基づき構成されるディレクトリの階層構造を管理する第２の管理手段と、

上記差分情報を選択的に取り込み、上記第２の管理手段で管理される上記ディレクトリの階層構造を変更する変更手段と

を有し、

上記コンテナエントリおよび／または上記リーフエントリには、最新の公開鍵証明情報を取得可能な情報と、該最新の公開鍵証明情報の失効情報とが格納されるようにしたことを特徴とする送受信システム。

【請求項１１】　請求項１０に記載の送受信システムにおいて、

上記失効情報は、上記公開鍵証明情報のシリアル番号であることを特徴とする送受信システム。

【請求項１２】　請求項１０に記載の送受信システムにおいて、

上記コンテナエントリおよび／または上記リーフエントリの属性に、上記最新公開鍵証明情報および上記最新の公開鍵証明情報を取得するための情報のうち何れか一方を選択して格納可能として送信するようにしたことを特徴とする送受信システム。

【請求項１３】　請求項１２に記載の送受信システムにおいて、

上記検出手段により上記差分が検出された更新事象からの時間に応じて、上記属性に格納される情報を、上記最新の公開鍵証明情報と上記最新の公開鍵証明情報を取得するための情報との間で変更可能なようにして送信することを特徴とする送受信システム。

【請求項１４】　請求項１０に記載の送受信システムにおいて、

上記変更手段は、上記公開鍵証明情報を取得するための証明情報パスに対応する上記コンテナエントリおよび／または上記リーフエントリの更新情報を上記選択的に取り込むことを特徴とする送受信システム。

【請求項１５】　公開鍵証明情報を階層的に管理するディレクトリの階層構造を送信し、送信されたディレクトリの階層構造を受信する送受信方法において、

自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、上記コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、上記コンテナエントリと上記リーフエントリとからなるディレクトリの階層構造を管理する第１の管理のステップと、

上記第１の管理のステップによって管理される上記ディレクトリの階層構造の変化を検出し、該検出結果に基づき上記ディレクトリの階層構造の変化の差分からなる差分情報を求める検出のステップと、

上記検出のステップで検出された上記差分情報を送信する送信のステップと、

上記送信のステップにより送信された、上記差分情報を受信する受信のステップと、

上記受信のステップにより受信された上記差分情報に基づき構成されるディレクトリの階層構造を管理する第２の管理のステップと、

上記差分情報を選択的に取り込み、上記第２の管理のステップで管理される上記ディレクトリの階層構造を変更する変更のステップと

を有し、

上記コンテナエントリおよび／または上記リーフエントリには、最新の公開鍵証明情報を取得可能な情報と、該最新の公開鍵証明情報の失効情報とが格納されるようにしたことを特徴とする送受信方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

この発明は、この発明は、階層構造を有すると共に、ネットワーク上に分散されて配置されたデータを一方向的に配信する系において、分散された公開鍵ディレクトリエントリの複製の更新を行うような送信装置および送信方法、受信装置および受信方法、ならびに、送受信システムおよび送受信方法に関する。

【０００２】

【従来の技術】

データの配信方法としては、種々の方法が提案されており、例えば、現在のインターネット上においては、ｈｔｔｐ(Hyper Text Transfer Protocol)のような、ＴＣＰ／ＩＰ(Transmission Control Protocol/Internet Protocol) を基本とするプロトコルが採用されている。ＴＣＰ／ＩＰでは、データの配信を受ける受信側からデータの送信側に対して発呼が行われ、さらに、データの送受信を行う毎に、送信側と受信側との間でコネクションが確立されるようになっている。そのため、信頼性の高いデータの配信を行うことができる。

【０００３】

その反面で、送信側やネットワークの負荷が大きくなり、効率的なデータ配信を行うことが困難になる場合があった。すなわち、データの提供を受ける端末が増加し、データを提供するサーバへのアクセスが集中すると、サーバやネットワークに多大な負荷がかかり、データを要求しても、そのデータを得るまでに多大な時間を要することがあった。

【０００４】

そこで、データの配信を、例えば、広い地域にわたって一斉同報が可能な衛星回線やＣＡＴＶ(Cable Television)回線、実用化が予定されている地上波ディジタル放送などを用いて行う方法が提案されている。この場合、端末の増加によって、サーバやネットワークに対する負荷が影響を受けることがない。

【０００５】

一方、近年では、インターネットなどディジタル通信ネットワークの発達により、ネットワーク上に膨大なデータが蓄積されるようになり、この膨大なデータを効率的に利用することが求められている。そこで、ネットワーク上に分散されて存在するデータを階層的に管理し、ユーザに提供する、ディレクトリサービス

が注目を集めている。ディレクトリサービスを利用することで、ユーザは、ネットワーク上に分散して存在する情報の中から、自分が必要な情報を素早く見つけ出し、アクセスすることが可能になる。

【０００６】

ディレクトリサービスは、例えば、国際標準であるＯＳＩ(Open Systems Interconnection)などによって、Ｘ．５００シリーズとして規定されている。Ｘ．５００によれば、ディレクトリについて、開放型システムの集合体であり、各開放型システムが協調して、現実世界のオブジェクトの集合に関する情報の、論理的なデータベースを持つと定義されている。

【０００７】

Ｘ．５００によるディレクトリサービスによれば、ディレクトリが擁する情報の検索や閲覧を行うことができる。また、例えば電話帳に例えられる一覧やユーザの認証などもディレクトリサービスで提供される。さらに、ディレクトリサービスでは、特に人間のユーザにとって覚え易く、推測ならびに認識し易いように、オブジェクトの名前が付けられる。

【０００８】

なお、このＸ．５００によるディレクトリサービスは、極めて包括的なものであって、プログラムサイズが非常に大きく、ＴＣＰ／ＩＰをプロトコルとするインターネットでは実現が非常に難しい。そのため、ＬＤＡＰ(Lightweight Directory Access Protocol) といった、ＴＣＰ／ＩＰ向けに軽量化されたディレクトリサービスが提案されている。

【０００９】

ディレクトリサービスをユーザが利用する場合、ディレクトリに対してフィルタ処理を行うことができる。フィルタ処理を行うフィルタリングマスクは、ユーザの、ディレクトリへのアクセスの傾向や嗜好に応じて設定される。例えば、ユーザの嗜好に応じて特定の情報ジャンルに対してフィルタリングマスクが設定される。ユーザは、フィルタリングマスクが設定されたディレクトリ情報を、選択的にアクセスすることができる。フィルタ処理を行うことにより、ユーザが不要な情報を保持する必要が無くなる。

【００１０】

近年では、上述した一斉同報を行うデータ伝送手段、すなわち、衛星回線やＣＡＴＶ回線、地上波ディジタル放送などで、このディレクトリサービスを行うことが提案されている。この場合には、ディレクトリサービスによる情報が一方向的に提供され、ユーザ側からの情報の要求ができない。したがって、ディレクトリサービスによるディレクトリ情報は、同一の情報が繰り返し送信される。

【００１１】

ユーザ側では、送信されてきた情報を、例えばテレビジョン受像機などに接続して用いられる、ディジタル放送用受信機であるＩＲＤ(Integrated Reciever Decoder) や、ＳＴＢ(Set Top Box) に蓄積する。このとき、上述したフィルタリングマスクを用いてフィルタ処理がなされ、ユーザの嗜好に応じたディレクトリ情報が選択的に蓄積される。フィルタ処理に用いられるフィルタリングマスクは、ユーザ側で設定される。

【００１２】

ところで、オープンなネットワークであるインターネットは、情報の盗聴、改竄、なりすましといった危険が常に伴う。これを防ぐのが認証システムであり、この認証システムを運用するための基盤となる環境を公開鍵基盤(ＰＫＩ: Public Key Infrastructure)と呼ぶ。ＰＫＩは、インターネット上の暗号化通信や電子メール、各種決済プロトコルなどを運用するための基盤環境として注目を集めており、その環境の信頼性が将来のインターネット上の商取引(EC: Electronic Commerce)の成否を左右するものと考えられている。

【００１３】

ＰＫＩで用いられる公開鍵暗号化方式は、秘密鍵および公開鍵の２つの鍵を用い、暗号化と復号化をこれら２つの別々の鍵で行う、非対称型の暗号化方式である。自分で保管する秘密鍵から公開鍵が作成され、公開鍵は、相手に渡すことができる。公開鍵で暗号化されたものは対応する秘密鍵でしか復号化できず、秘密鍵で暗号化されたものは対応する公開鍵でしか復号化できない。この特性を利用して、電子署名を行うことができる。

【００１４】

また、公開鍵だけでは上述の「なりすまし」が可能なので、認証局（ＣＡ:Certificate Authority）による公開鍵証明書（ディジタル証明書）の発行を以て、公開鍵の管理とその認証が行われる。公開鍵証明書の発行は、例えば以下のようにしてなされる。加入者が秘密鍵と公開鍵のペアを所定の方法で作成し、その公開鍵を認証局に提出する。認証局は、加入者の身分などを確認してから、証明書のシリアル番号、加入者の名前、公開鍵、有効期間などの情報に認証局の署名をし、認証局が予め公開している公開鍵を利用して電子署名を行った、公開鍵証明書を発行する。

【００１５】

図２８を用いて、公開鍵証明書の仕組みについて説明する。図２８Ａは、公開鍵証明書の作成の手順を概略的に示す。公開鍵証明書は、シリアル番号、発行者（認証局）名、有効期限、所有者名などの情報（ここでは証明書情報と称する）および所有者の公開鍵と、証明書情報をダイジェスト関数で変換して得られた「指紋」を認証局の秘密鍵で暗号化した認証局の署名とからなる。なお、ダイジェスト関数は、長い文書を短い決まった長さに変換する、元の情報が少しでも変われば変換結果が大きく異なる、という特徴を持つ関数である。このことから、ダイジェスト関数で得られた値は、元の文書の「指紋」と称される。

【００１６】

図２８Ｂは、このようにして作成された公開鍵証明書を確認する手順を概略的に示す。公開鍵証明書に含まれる証明書情報からダイジェスト関数により「指紋」が求められる。一方、公開鍵証明書において、証明書情報に添付される署名が認証局の公開鍵で復号化される。この復号化された結果と、上述の「指紋」とが比較され、両者が一致すれば、公開鍵証明書が改竄されていない正当なものであると判断することができる。

【００１７】

【発明が解決しようとする課題】

現在、ＰＫＩの重要な機能と位置付けられる、上述の公開鍵証明書を、広範囲に分散したユーザに対して配布管理するシステムの完備が急がれている。しかしながら、従来では、失効した公開鍵証明書の通知を、広範囲に分散した大多数の

ユーザに効果的に通知する方式が確立されていないという問題点があった。

【００１８】

例えば、証明書を盗難された場合や、個人が退職してその証明書を利用する資格を失った場合など、公開鍵証明書に定められている有効期限を待たずに、その公開鍵証明書を無効化したいことがある。

【００１９】

このような場合、無効化された（失効した）証明書のシリアル番号や失効した日付などを、公開鍵証明書を配布・管理するシステムに登録しておき、認証手続きの際に公開鍵証明書を利用するアプリケーションに対して、対象の公開鍵証明書が失効していないかを毎回確認させるようにしなければならない。この公開鍵証明書の失効情報、さらに公開鍵証明書そのものを配布、管理および失効照会を提供するシステムとして、オンライン失効情報照会（ＰＫＩディレクトリサービス）サービスが考えられる。

【００２０】

このＰＫＩディレクトリサービスにおいて、クライアント（証明書の失効情報を照会するアプリケーション）の数が多数になると、問い合わせの負荷の増大でシステムが破綻してしまうおそれがある。そのため、複数の分散されたＰＫＩディレクトリサーバに大元のＰＫＩディレクトリサーバの複製を作り、問い合わせの負荷の分散を図る方法がとられる。しかしながら、複製を作った場合には、それらを最新の情報に更新管理する必要が生じるが、複製の数の規模が非常に大きくなると、その更新管理が難しくなるという問題点があった。

【００２１】

したがって、この発明の目的は、ＰＫＩディレクトリサーバからＰＫＩディレクトリクライアントへ公開鍵証明書の失効情報の通知を、効率よく行うことができる送信装置および送信方法、受信装置および受信方法、ならびに、送受信システムおよび送受信方法を提供することにある。

【００２２】

【課題を解決するための手段】

この発明は、上述した課題を解決するために、公開鍵証明情報を階層的に管理

するディレクトリの階層構造を送信する送信装置において、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造を管理する管理手段と、管理手段によって管理されるディレクトリの階層構造の変化を検出し、検出結果に基づきディレクトリの階層構造の変化の差分からなる差分情報を求める検出手段と、検出手段で検出された差分情報を送信する送信手段とを有し、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とを格納するようにしたことを特徴とする送信装置である。

【００２３】

また、この発明は、公開鍵証明情報を階層的に管理するディレクトリの階層構造を送信する送信方法において、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造を管理する管理のステップと、管理のステップによって管理されるディレクトリの階層構造の変化を検出し、検出結果に基づきディレクトリの階層構造の変化の差分からなる差分情報を求める検出のステップと、検出のステップで検出された差分情報を送信する送信のステップとを有し、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とを格納するようにしたことを特徴とする送信方法である。

【００２４】

また、この発明は、送信された、公開鍵証明情報を階層的に管理するディレクトリの階層構造を受信する受信装置において、送信された、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造の変化を検出した検出結果に基づき求められたディレクトリの階層構

造の変化の差分からなる差分情報を受信する受信手段と、受信手段により受信された差分情報に基づき構成されるディレクトリの階層構造を管理する管理手段と、差分情報を選択的に取り込み、管理手段で管理されるディレクトリの階層構造を変更する変更手段とを有し、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とが格納されて送信されることを特徴とする受信装置である。

【００２５】

また、この発明は、送信された、公開鍵証明情報を階層的に管理するディレクトリの階層構造を受信する受信方法において、送信された、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造の変化を検出した検出結果に基づき求められたディレクトリの階層構造の変化の差分からなる差分情報を受信する受信のステップと、受信のステップにより受信された差分情報に基づき構成されるディレクトリの階層構造を管理する管理のステップと、差分情報を選択的に取り込み、管理のステップで管理されるディレクトリの階層構造を変更する変更のステップとを有し、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とが格納されて送信されることを特徴とする受信方法である。

【００２６】

また、この発明は、公開鍵証明情報を階層的に管理するディレクトリの階層構造を送信し、送信されたディレクトリの階層構造を受信する送受信システムにおいて、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造を管理する第１の管理手段と、第１の管理手段によって管理されるディレクトリの階層構造の変化を検出し、検出結果に基づきディレクトリの階層構造の変化の差分からなる差分情報を求める検出手段と

、検出手段で検出された差分情報を送信する送信手段と、送信手段により送信された、差分情報を受信する受信手段と、受信手段により受信された差分情報に基づき構成されるディレクトリの階層構造を管理する第２の管理手段と、差分情報を選択的に取り込み、第２の管理手段で管理されるディレクトリの階層構造を変更する変更手段とを有し、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とが格納されるようにしたことを特徴とする送受信システムである。

【００２７】

また、この発明は、公開鍵証明情報を階層的に管理するディレクトリの階層構造を送信し、送信されたディレクトリの階層構造を受信する送受信方法において、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造を管理する第１の管理のステップと、第１の管理のステップによって管理されるディレクトリの階層構造の変化を検出し、検出結果に基づきディレクトリの階層構造の変化の差分からなる差分情報を求める検出のステップと、検出のステップで検出された差分情報を送信する送信のステップと、送信のステップにより送信された、差分情報を受信する受信のステップと、受信のステップにより受信された差分情報に基づき構成されるディレクトリの階層構造を管理する第２の管理のステップと、差分情報を選択的に取り込み、第２の管理のステップで管理されるディレクトリの階層構造を変更する変更のステップとを有し、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とが格納されるようにしたことを特徴とする送受信方法である。

【００２８】

上述したように、請求項１および５に記載の発明は、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階

層構造が管理され、管理されたディレクトリの階層構造の変化の検出結果に基づき求められた、ディレクトリの階層構造の変化の差分からなる差分情報が送信され、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とが格納されるため、受信側では、受信側が有している公開鍵証明情報が最新のものであるか否かを知ることができると共に、受信側が有している公開鍵情報が最新のものでないとされた場合には、最新の公開鍵証明情報を取得することができる。

【００２９】

また、請求項６および９に記載の発明は、送信された、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造の変化を検出した検出結果に基づき求められたディレクトリの階層構造の変化の差分からなる差分情報を受信し、受信された差分情報に基づき構成されるディレクトリの階層構造が、選択的に取り込まれた差分情報に基づき変更されると共に、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とが格納されて送信されているため、公開鍵証明情報が最新のものであるか否かを知ることができると共に、公開鍵情報が最新のものでないとされた場合には、最新の公開鍵証明情報を取得することができる。

【００３０】

また、請求項１０および１５に記載の発明は、送信側では、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下にあって、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造の変化を検出し、検出結果に基づき求められた、ディレクトリの階層構造の変化の差分からなる差分情報を送信し、受信側では、送信された差分情報を受信し、受信された差分情報に基づき構成されるディレクトリの階層構造が選択的に取り込まれた差分情報により変更されると共に、コンテナエントリおよ

び／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とが格納されて送信されるため、受信側では、受信側が有している公開鍵証明情報が最新のものであるか否かを知ることができると共に、受信側が有している公開鍵情報が最新のものでないとされた場合には、最新の公開鍵証明情報を取得することができる。

【００３１】

【発明の実施の形態】

以下、この発明の実施の一形態について説明する。先ず、理解を容易とするために、この発明の実施の一形態の説明に先んじて、この発明に適用できるディレクトリサービスについて説明する。図１は、この発明に適用できる系の一例を示す。送信側１は、例えばインターネットや放送ネットワークなどの、図示されないネットワーク上に存在する多数のコンテンツをツリー状の階層構造に整理し、ディレクトリ構造として管理している。送信側１では、このディレクトリ構造を示すディレクトリ情報を放送ネットワーク２に対して送信する。受信側３は、図２に一例が示されるように、放送ネットワーク２に対して多数が接続され、放送ネットワーク２を介してなされた放送をそれぞれが受信可能とされている。受信側３は、放送ネットワーク２で放送されたディレクトリ情報を受信し、受信されたディレクトリ情報を参照して、放送ネットワーク２や他のネットワーク上に存在する多数の情報の中から必要な情報を選択し、入手することができる。

【００３２】

図１に一例が示されるように、送信側１は、送信側ディレクトリサービスクライアント１０（以下、送信側クライアント１０と略す）、送信側ディレクトリサーバ１１（以下、送信側サーバ１１と略す）および送信側ディレクトリサーバレプリケータ１２（以下、送信側レプリケータ１２と略す）からなる。これら送信側クライアント１０、送信側サーバ１１および送信側レプリケータ１２は、互いに例えばインターネットといったネットワークなどで接続されており、相互に通信が行われる。

【００３３】

送信側クライアント１０は、例えば図示されないネットワークなどによってコ

ンテンツを提供するコンテンツプロバイダであって、ディレクトリ構造の変更や更新を行う。送信側クライアント１０は、ネットワーク上の何処に位置していてもよい。送信側サーバ１１は、送信側クライアント１０の内容照会や変更などを行い、ディレクトリ構造を管理する。送信側サーバ１１は、ネットワーク上で分散して構成することができる。送信側レプリケータ１２は、送信側サーバ１１で管理されているディレクトリ構造をモニタしてディレクトリ構造の更新を検出する。そして、送信側レプリケータ１２は、この検出結果に基づき、更新前の構造と更新後の構造とを比較して差分を抽出し、ディレクトリ構造の差分更新情報を構成する。差分更新情報は、放送ネットワーク２に送信される。差分更新情報の構成については、後述する。

【００３４】

受信側３は、受信側ディレクトリサーバレプリケータ１７（以下、受信側レプリケータ１７と略す）、受信側ディレクトリサーバ１６（以下、受信側サーバ１６と略す）および受信側ディレクトリサービスクライアント１５（以下、受信側クライアント１５と略す）からなる。受信側３は、例えばパーソナルコンピュータや従来技術で述べたＳＴＢ、ＩＲＤなどであり、受信側クライアント１５は、ディレクトリ構造にアクセスして複数の異なる形式のデータの取得ならびに表示ができるようにされた、例えばWWW(World Wide Web)ブラウザなどのアプリケーションソフトウェアである。また、受信側サーバ１６は、ローカルなデータベースからなり、ディレクトリ情報が格納される。

【００３５】

放送ネットワーク２で送信されたディレクトリ情報、ディレクトリ構造の更新情報および更新情報の差分情報などは、受信側レプリケータ１７に受信される。受信側レプリケータ１７では、受信されたこれらの情報に基づき、受信側サーバ１６に格納されたデータベースを更新し、ディレクトリ構造の再構築を行う。受信側クライアント１５は、例えばユーザの指示により、受信側レプリケータ１７に対して必要な情報を要求する。受信側レプリケータ１７は、この要求に基づき受信側サーバ１６のデータベースを検索して、受信側クライアント１５に対して、例えば要求された情報のアドレスを返す。受信側クライアント１５は、このア

ドレスに基づき、例えば図示されないネットワーク上に存在する情報にアクセスすることができる。

【００３６】

図３を用いて、ディレクトリ構造について説明する。ディレクトリは、図に示されるように、ツリー状の階層構造からなる。ツリーの各節点（ノード）をエントリと称し、各エントリには、情報が格納される。エントリには、ルートエントリ、コンテナエントリおよびリーフエントリの３種類が定義される。コンテナエントリは、さらに配下のエントリを包含することができるエントリである。コンテナエントリによって構成される階層を、以下、コンテナ階層と称する。

【００３７】

コンテナエントリ以外のエントリを、リーフエントリと称する。リーフエントリは、配下にエントリを含むことができない、末端の節点である。リーフエントリによる階層を、以下、リーフ階層と称する。リーフ階層は、コンテナエントリの配下に構成される。

【００３８】

また、ディレクトリツリーの最上位のエントリは、ルートエントリと称され、当該ディレクトリ構造で完結される世界全体を示すエントリである。なお、以下では、コンテナエントリは、少なくとも一つのリーフエントリあるいはコンテナエントリを配下に持つものとする。

【００３９】

エントリは、複数の属性を持つ。エントリが持つ属性のうちで、ディレクトリツリーで一意に識別される名前を、エントリ名と称する。エントリ名によって、そのエントリの、ディレクトリ構造上の位置を特定することができる。図３の例では、ルートエントリには、エントリ名Ａが与えられている。ルートエントリの直接的な配下のエントリには、図の左側に示されるリーフエントリにはエントリ名Ａ．Ｂが、右側のコンテナエントリにはエントリ名Ａ．Ｃがそれぞれ与えられる。以下、ルートエントリから階層構造を辿った順にピリオドで区切られて、各エントリに対してエントリ名が与えられる。

【００４０】

図４は、コンテナエントリの構造の一例を示す。コンテナエントリは、コンテナエントリ自身の属性と、自分の配下のコンテナエントリおよびリーフエントリのリストとを有する。配下のエントリのリストは、要素を含まないこともできる。また、コンテナエントリ自身の属性は、図示されるように、複数持つことができる。コンテナエントリの属性は、図４Ｂに示されるように、属性名と属性値とから構成される。

【００４１】

図５は、リーフエントリの構造の一例を示す。リーフエントリは、図５Ａに示されるように、リーフエントリの属性を複数、有する。図５Ｂは、リーフエントリの属性のより具体的な例である。各属性は、属性名と属性値とからなる。例えばリーフエントリがコンテンツの検索情報である場合には、属性名の一つにアドレスがあり、その属性値として、インターネット上の場所を示すＵＲＬ(Uniform Resource Locator)などの、コンテンツのアドレス情報が記述される。

【００４２】

ディレクトリ構造は、例えば、そのディレクトリ構造で完結する世界全体を表すルートエントリの下に、コンテナエントリが例えば情報ジャンルに応じてツリー状に分類され配されて、構成される。

【００４３】

送信側１の構成について、より具体的に説明する。送信側サーバ１１には、図３～図５で既に説明したような構成に準えて、ディレクトリ構造が管理されている。送信側クライアント１０は、提供するコンテンツに応じて、送信側サーバ１１で管理されているディレクトリ構造に対して変更などを加える。送信側サーバ１１に対してなされた変更は、送信側レプリケータ１２にモニタされる。

【００４４】

図６は、送信側レプリケータ１２の機能を説明するための機能ブロック図である。送信側レプリケータ１２は、例えば一般的なコンピュータシステムで構成することができ、ＣＰＵ(Central Processing Unit)、メモリ、ハードディスクといった記録および記憶媒体、通信手段、タイマ、ユーザインターフェイスなどからなる。この図６に示される機能ブロックは、ＣＰＵ上で動作するアプリケーシ

ョンソフトウェアにより実現され、各モジュールは、アプリケーションソフトウェア上の機能的な単位であって、それぞれがソフトウェアからなる。

【００４５】

送信側レプリケータ１２は、更新検知モジュール２０、メッセージ生成モジュール２１およびメッセージ放送モジュール２２からなる。これらモジュール２０、２１および２２のそれぞれは、コンテナ階層に関する処理を行うモジュールと、リーフ階層に関する処理を行うモジュールとを有する。

【００４６】

更新検知モジュール２０は、送信側サーバ１１を参照して、サーバ１１上に管理されているディレクトリ構造に変化があったかどうかを検知するモジュールで、コンテナ階層更新検知モジュール２３とリーフ階層更新検知モジュール２４とからなる。コンテナ階層更新検知モジュール２３は、送信側サーバ１１をモニタして、コンテナ階層の構造の変化を検知する。また、リーフ階層更新検知モジュール２４は、送信側サーバ１１をモニタして、リーフ階層の構造およびリーフエントリの内容の変化を検知する。

【００４７】

メッセージ生成モジュール２１は、更新検知モジュール２０によるディレクトリ構造の変化の検知結果に基づく、ディレクトリ構造の差分更新情報が示されたメッセージを生成するモジュールで、コンテナストラクチャアップデートメッセージ生成モジュール２５と、リーフエントリアップデートメッセージ生成モジュール２６とからなる。コンテナストラクチャアップデートメッセージ生成モジュール２５は、コンテナ階層更新検知モジュール２３の検知結果に基づき、コンテナ階層における構造変化に関する差分更新情報が示されたメッセージを生成する。また、リーフエントリアップデートメッセージ生成モジュール２６は、リーフ階層更新検知モジュール２４の検知結果に基づき、リーフ階層における更新情報が示されたメッセージを生成する。

【００４８】

メッセージ放送モジュール２２は、メッセージ生成モジュール２１で生成されたメッセージを放送ネットワーク２に対して放送するモジュールで、コンテナス

トラクチャアップデートメッセージ放送モジュール２７とリーフエントリアップデートメッセージ放送モジュール２８とからなる。コンテナストラクチャアップデートメッセージ放送モジュール２７は、上述した、コンテナストラクチャアップデートメッセージ生成モジュール２５で生成されたメッセージを放送する。また、リーフエントリアップデートメッセージ放送モジュール２８は、上述した、リーフエントリアップデートメッセージ生成モジュール２６で生成されたメッセージを放送する。なお、メッセージ放送モジュール２２からの放送ネットワーク２に対するメッセージの放送は、同一のメッセージが所定回数だけ、サイクリックに送信されて行われる。

【００４９】

次に、受信側３の構成について、より具体的に説明する。図７は、受信側クライアント１５の機能を説明するための機能ブロック図である。受信側クライアント１５は、例えば一般的なコンピュータシステムで構成することができ、ＣＰＵ(Central Processing Unit)、メモリ、ハードディスクといった記録および記憶媒体、通信手段、ユーザインターフェイスなどからなる。この図７に示される機能ブロックは、ＣＰＵ上で動作するアプリケーションソフトウェアにより実現され、各モジュールは、アプリケーションソフトウェア上の機能的な単位であって、それぞれがソフトウェアからなる。

【００５０】

受信側クライアント１５は、上述したように、例えばＷＷＷブラウザであり、供給されたコンテンツ、例えば画像データ、テキストデータ、音声データおよび動画データを統一的に表示および再生することができるようにされている。また、所定の入力手段を用いてユーザによって入力された指示に基づき、上述の表示ならびに再生を制御することができる。

【００５１】

受信側クライアント１５は、ディレクトリ検索モジュール３０、ユーザ対話管理モジュール３１およびコンテンツ取得モジュール３２とからなる。また、ユーザ対話管理モジュール３１に対して、例えばキーボードなどのテキスト入力手段、マウスなどのポインティングデバイスや表示装置などからなるユーザインター

フェイス３３が接続される。ユーザの、受信側クライアント１５に対するコンテンツ検索要求の入力は、ユーザインターフェイス３３を用いて、ユーザ対話モジュール３１に対して対話形式で行われる。

【００５２】

ユーザ対話モジュール３１に対してコンテンツ検索要求が入力されると、ユーザ対話管理モジュール３１からディレクトリ検索モジュール３０に対して、コンテンツのアドレスを検索するため、コンテンツに対応するディレクトリエントリを検索するよう依頼が出される。ディレクトリ検索モジュール３０では、この検索依頼に応じて、受信側サーバ１６に対してディレクトリエントリ検索要求を送る。

【００５３】

受信側サーバ１６での検索要求に基づくディレクトリエントリの検索結果がディレクトリ検索モジュール３０に返される。検索結果は、ディレクトリ検索モジュール３０からユーザ対話管理モジュール３１に返される。そして、検索結果のディレクトリエントリ情報から、例えば検索結果がリーフエントリであれば、属性の一つであるコンテンツのアドレス情報が取り出される。ユーザ対話管理モジュール３１からコンテンツ取得モジュール３２に対して、取り出されたアドレス情報によって示されるコンテンツを取得するよう、コンテンツ取得依頼が出される。

【００５４】

コンテンツ取得モジュール３２では、受け取ったコンテンツ取得依頼に基づき、コンテンツサーバ３５に対してコンテンツの取得要求を送る。コンテンツサーバ３５は、受信側クライアント１５と、例えばインターネットといった双方向ネットワーク３６を介して接続されるサーバであり、ユーザに対してコンテンツを提供する。コンテンツの提供は、双方向ネットワーク３６を介して行ってもいいし、放送ネットワーク２によって行うこともできる。

【００５５】

コンテンツ取得要求に基づきコンテンツサーバ３５から取得されたコンテンツは、例えば双方向ネットワーク３６を介してコンテンツ取得モジュール３２に供

給され、コンテンツ取得モジュール３２からユーザ対話管理モジュール３１に返される。ユーザ対話管理モジュール３１では、受け取ったコンテンツをユーザインターフェイス３３に出力する。

【００５６】

なお、要求するコンテンツが放送ネットワーク２で伝送されるものである場合には、コンテンツ取得モジュール３２は、放送ネットワーク２で放送される所望のコンテンツを、コンテンツ取得依頼に基づき直接的に取得するようにしてもよい。

【００５７】

図８は、受信側サーバ１６の機能を説明するための機能ブロック図である。この受信側サーバ１６も、説明は省略するが、上述の受信側クライアント１５と同様に、一般的なコンピュータシステムによって構成される。受信側サーバ１６は、ディレクトリ更新要求処理モジュール４０、ディレクトリデータベース４１およびディレクトリ検索要求処理モジュール４２からなる。

【００５８】

ディレクトリデータベース４１は、送信側サーバ１１で管理されるディレクトリ構造に基づくディレクトリ情報が格納されている。上述したように、受信側レプリケータ１７は、放送ネットワーク２を介して送信側１から送信されたディレクトリ構造の差分更新情報を受信する。詳細は後述するが、受信側レプリケータ１７からディレクトリ更新要求処理モジュール４０に対して、差分更新情報に基づきディレクトリデータベース４１に格納されているディレクトリ情報の更新を行うよう、要求が出される。ディレクトリ更新要求処理モジュール４０では、この要求に基づき、差分更新情報を用いてディレクトリデータベース４１に格納されているディレクトリ情報の更新を行う。

【００５９】

一方、上述した受信側クライアント１５からのディレクトリエントリの検索要求は、ディレクトリ検索要求処理モジュール４０に受け取られる。ディレクトリ検索要求処理モジュール４０によって、受け取った検索要求に基づきディレクトリデータベース４１が検索される。検索した結果得られたディレクトリエントリ

、例えばリーフエントリ中のアドレス情報は、ディレクトリ検索要求処理モジュール４２から受信側クライアント１５に返される。

【００６０】

上述したように系を構成することで、ユーザは、受信側クライアント１５によってディレクトリ情報を検索し、検索結果として、所望のコンテンツが提供されるアドレス情報を得ることができる。そして、ユーザは、得られたアドレス情報に基づき所望のコンテンツを取得することができる。また、ディレクトリ構造は、常に送信側レプリケータ１２によってモニタされ、ディレクトリ構造に変更が生じたときには、変更に伴うディレクトリ構造の差分更新情報が放送ネットワーク２を介して受信側レプリケータ１７に供給される。ユーザ側では、供給された差分更新情報に基づき、受信側レプリケータ１７によってディレクトリデータベース４１に格納されたディレクトリ情報が変更される。そのため、ユーザは、常に実際のディレクトリ構造と同期したディレクトリ情報を、ディレクトリデータベース４１に保持することができる。

【００６１】

次に、図９および図１０を用いて、ディレクトリ構造の変化に伴い生成される、ディレクトリ構造の差分更新情報について説明する。以下では、スキーマバージョンＳｖで特定されるあるコンテナ階層のコンテナエントリＸの配下に、コンテナエントリＣまたはリーフエントリｌを追加または削除する処理を、

（Ｓｖ，Ｘ，〔＋／－〕〔Ｃ／ｌ〕）

と記述する。このディレクトリ構造に対する処理を示す記述は、この記述による処理で変化したディレクトリ構造の、元の構造との差分を表しており、これを差分更新情報として用いることができる。

【００６２】

スキーマバージョンＳｖは、ディレクトリ構造の変更に伴い変化する値である。コンテナエントリＸ（またはＣ）は、コンテナエントリ名であり、ここでは大文字のアルファベットで表す。リーフエントリｌは、リーフエントリ名であり、ここでは小文字のアルファベットで表す。エントリの追加は〔＋〕で表され、削除は〔－〕で表される。括弧〔〕中のスラッシュ記号は、その両側に記された何

方かが記述されることを示す。なお、図９および図１０において、二重線の四角は、コンテナエントリを表し、単線の四角は、リーフエントリを表す。ルートエントリは、接続線だけが示され、本体は省略されている。

【００６３】

図９Ａにおいて、図示されないルートエントリの配下にコンテナエントリＡのみが存在している。この状態をスキーマバージョンＳｖ＝１とする。この状態に、上述の記述に従い（１，Ａ，＋Ｂ）の処理を行う。すなわち、コンテナエントリＡの配下にコンテナエントリＢが加えられる。すると、図９Ｂのようなディレクトリ構造になる。図９Ａの状態にコンテナエントリＢが加えられたことで、コンテナエントリの階層イメージが変化したとされ、スキーマバージョンＳｖ＝２とされる。

【００６４】

図９Ｂの状態に、さらに（２，Ａ，＋ａ）の処理を行う。すなわち、コンテナエントリＡの配下にリーフエントリａを加える。すると、図９Ｃのようなディレクトリ構造になる。さらにまた、図９Ｃの状態に（２，Ａ，－ａ）の処理を行う。すなわち、リーフエントリａを、コンテナエントリＡの配下から削除する。すると、図９Ｄのようなディレクトリ構造になる。さらに、図９Ｄの状態に（２，Ａ，－Ｂ）の処理を行う。すなわち、コンテナエントリＢを、コンテナエントリＡの配下から削除する。すると、図９Ｅのようなディレクトリ構造になる。

【００６５】

図９Ｅの状態では、コンテナエントリの階層イメージが図９Ｄの状態から変化しているため、スキーマバージョンＳｖが更新され、スキーマバージョンＳｖ＝３になる。したがって、図９Ｅの状態において、コンテナエントリＡの配下にコンテナエントリＣを加える処理は、（３，Ａ，＋Ｃ）と記述できる。この処理を行うと、図９Ｆに示されるディレクトリ構造となる。

【００６６】

この図９の例では、（１，Ａ，＋Ｂ）、（２，Ａ，＋ａ）、（２，Ａ，－ａ）、（２，Ａ，－Ｂ）および（３，Ａ＋Ｃ）がそれぞれの段階での差分更新情報とされる。

【００６７】

図１０は、ディレクトリ構造を変更する別の例を示す。上述の図９に示される例では、一度に一つの処理を行っていたが、図１０では、２つずつの処理をまとめて行っている。図１０Ａは、図示されないルートエントリの配下にコンテナエントリＡのみが存在している。この状態がスキーマバージョンＳｖ＝１とする。この図１０Ａの状態に対して、（１，Ａ，＋Ｂ）および（１，Ａ，＋ａ）の処理を順に行う。すなわち、コンテナエントリＡの配下に、コンテナエントリＢとリーフエントリａとが追加される。すると、図１０Ｂのようなディレクトリ構造となる。コンテナエントリの階層イメージが変わり、スキーマバージョンＳｖ＝２となる。

【００６８】

図１０Ｂの状態に、さらに、（２，Ａ，－ａ）および（２，Ｂ，＋ｂ）の処理を順に行う。すなわち、コンテナエントリＡの配下からリーフエントリａを削除し、その後、コンテナエントリＢの配下にリーフエントリｂを追加する。すると、図１０Ｃのようなディレクトリ構造となる。

【００６９】

図１０Ｃの状態に対して、さらに、（２，Ｂ，＋Ｃ）および（２，Ｃ，＋ｃ）の処理を行う。すなわち、コンテナエントリＢの配下にコンテナエントリＣを追加した後に、追加されたコンテナエントリＣの配下にリーフエントリｃを追加する。この処理においては、追加されたコンテナエントリに対してさらにエントリが追加されるため、処理の前後を入れ換えることができない。処理後、図１０Ｄのようなディレクトリ構造となり、コンテナエントリの階層イメージが変わったため、スキーマバージョンＳｖが更新され、スキーマバージョンＳｖ＝３とされる。

【００７０】

この図１０の例では、（１，Ａ，＋Ｂ）および（１，Ａ，＋ａ）、（２，Ａ，－ａ）および（２，Ｂ，＋ｂ）、ならびに、（２，Ｂ，＋Ｃ）および（２，Ｃ，＋ｃ）がそれぞれの段階での差分更新情報とされる。上述したように、複数の処理を一つのディレクトリ構造の更新処理としてまとめて行うときには、処理の順

序を考慮する必要がある。

【００７１】

なお、ディレクトリ構造の差分更新情報は、上述の例に限定されず、適用されるシステムに応じて様々な形式とすることができる。

【００７２】

また、リーフエントリに関しては、コンテナエントリの配下からの削除やコンテナエントリの配下への追加の他に、内容の修正だけが行われることがある。内容の修正だけが行われた場合には、ディレクトリ構造における変化は、生じない。この場合には、例えば、リーフエントリ名と、当該リーフエントリ中で修正のあった属性名および属性値の列で、差分更新情報が構成される。一例として、

```
{
    LeafEntryName,
    Set of { AttributeName,AttributeValue }
}
```

このように差分更新情報が記述される。

【００７３】

この発明が適用される系では、上述したように、差分更新情報は、送信側１から受信側３に対して放送ネットワーク２を介して一方向的に送信される。また、一つの送信側１に対して複数の受信側３が存在し、複数の受信側３の稼働状態もそれぞれ異なる。そのため、送信側１で管理されているディレクトリ情報と、受信側３で管理されているディレクトリ情報とを同期させる必要がある。

【００７４】

以下、送信側１の送信側サーバ１１に格納されたディレクトリ情報と、受信側３の受信側サーバ１６に格納されたディレクトリ情報とを同期させ、ディレクトリ構造の同期管理方法について説明する。

【００７５】

最初に、図１１のフローチャートを用いてコンテナエントリの同期管理方法について説明する。先ず、ステップＳ１で、送信側クライアント１０によって、送信側サーバ１１で管理されているディレクトリ構造の、コンテナ階層の構成が変

更される。例えば、コンテナエントリの配下に新たなコンテナエントリやリーフエントリが追加される処理や、コンテナエントリの配下のコンテナエントリやリーフエントリが削除される処理が行われる。

【００７６】

次のステップＳ２では、送信側サーバ１１に対してなされた変更が送信側レプリケータ１２によって検知され、検知結果に基づき、コンテナ階層構成の変更によるコンテナ構造更新情報Ｍｓｇ．１が生成される。生成されたコンテナ構造更新情報Ｍｓｇ．１は、放送ネットワーク２に対して放送される。コンテナ構造更新情報Ｍｓｇ．１の放送は、同一の内容が所定回数、サイクリックに繰り返されて行われる。

【００７７】

放送されたコンテナ構造更新情報Ｍｓｇ．１は、ステップＳ３で、受信側レプリケータ１７によって受信される。受信側レプリケータ１７は、受信側サーバ１６に格納されたディレクトリ情報に管理されるコンテナ階層構成を、受信したコンテナ構造更新情報Ｍｓｇ．１に基づき変更する。これにより、送信側１と受信側３とで、ディレクトリ情報のコンテナ階層の構造の同期がとられる。

【００７８】

コンテナ構造更新情報Ｍｓｇ．１のフォーマットは、例えば、

```
Container Structure Update Message {
    MessageID,
    差分更新情報
}
```

このように定義される。「MessageID 」（メッセージＩＤ）は、このメッセージ（コンテナ構造更新情報Ｍｓｇ．１）の識別情報であって、例えば、このメッセージが生成される毎に１ずつ増加される整数である。また、「差分更新情報」は、コンテナ階層構成の変更による、上述したディレクトリ構造の差分更新情報である。

【００７９】

図１１のフローチャートにおけるステップＳ２の処理を、図１２のフローチャ

ートを用いて、より詳細に説明する。この図１２のフローチャートによる処理は、全て送信側レプリケータ１２上で行われる。先ず、ステップＳ１０で、送信側サーバ１１上のコンテナエントリの階層構成の情報が全て読み込まれる。読み込まれたコンテナエントリの階層構成の情報は、送信側レプリケータ１２が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー１として記憶される。

【００８０】

コピー１が記憶されたら、次のステップＳ１１で、タイマが所定の時間にセットされ、起動される。ステップＳ１２によって、タイマにセットされた所定時間を超過したかどうかが判断され、所定時間を超過したと判断されたなら、処理はステップＳ１３に移行する。ステップＳ１３では、再び送信側サーバ１１上のコンテナエントリの階層構成の情報が全て読み込まれる。読み込まれたコンテナエントリの階層構造は、送信側レプリケータ１２が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー２として記憶される。

【００８１】

次のステップＳ１４では、ステップＳ１０で記憶されたコピー１と、ステップＳ１３で記憶されたコピー２とが比較される。比較の結果、両者に差分が無いとされれば（ステップＳ１５）、処理はステップＳ１１へ戻され、再びタイマがセットされ、コピー２の記憶がなされる。

【００８２】

一方、ステップＳ１４で、コピー１とコピー２との間に差分があるとされれば、処理はステップＳ１６に移行する。ステップＳ１６では、コピー１とコピー２との差分に基づき、差分更新情報が生成される。そして、この差分更新情報が記述されたコンテナ構造更新情報Ｍｓｇ．１が生成される。生成されたコンテナ構造更新情報Ｍｓｇ．１は、放送ネットワーク２に対して送信され、放送される。放送されたコンテナ構造更新情報Ｍｓｇ．１は、受信側レプリケータ１７に受信される。

【００８３】

ステップＳ１６でコンテナ構造更新情報Ｍｓｇ．１が放送されると、次のステ

ップＳ１７で、コピー１の内容がコピー２の内容で置き替えられ、処理はステップＳ１１に戻される。

【００８４】

図１１のフローチャートにおけるステップＳ３の処理を、図１３のフローチャートを用いて、より詳細に説明する。この図１３のフローチャートによる処理は、全て受信側レプリケータ１７上で行われる。最初のステップＳ２０で、送信側レプリケータ１２によって放送ネットワーク２を介して放送されたコンテナ構造更新情報Ｍｓｇ．１が、受信側レプリケータ１７によって受信される。

【００８５】

ステップＳ２１で、ステップＳ２０での受信がコンテナ構造更新情報Ｍｓｇ．１の初回の受信かどうかが判断される。そして、この受信が初回の受信では無いと判断されたら、処理はステップＳ２３に移行し、受信されたコンテナ構造更新情報Ｍｓｇ．１のメッセージＩＤが受信側レプリケータ１７が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー３として記憶される。

【００８６】

次のステップＳ２４では、受信されたコンテナ構造更新情報Ｍｓｇ．１の内容、すなわち、コンテナ構造更新情報Ｍｓｇ．１に記述された差分更新情報に基づき、受信側サーバ１６で管理されているディレクトリ情報が更新され、そのディレクトリ情報で示されるコンテナ階層の構成が変更される。ステップＳ２４の処理の後、処理はステップＳ２０に戻される。

【００８７】

一方、上述のステップＳ２１で、ステップＳ２０でのコンテナ構造更新情報Ｍｓｇ．１の受信が初回の受信では無いと判断されたら、処理はステップＳ２２に移行する。ステップＳ２２では、受信されたコンテナ構造更新情報Ｍｓｇ．１に記述されるメッセージＩＤと、前回の受信の際のステップＳ２３の処理でコピー３として記憶されたメッセージＩＤとが同一であるかどうかが判断される。若し、同一であるとされれば、処理はステップＳ２０に戻される。

【００８８】

一方、ステップＳ２２で、両者のメッセージＩＤが同一では無いとされれば、処理はステップＳ２３に移行する。ステップＳ２３では、上述したように、メッセージＩＤが記憶媒体にコピー３として記憶される。この場合には、新たに受信されたメッセージＩＤで、前回受信され記憶されたメッセージＩＤが例えば上書きされることになる。そして、次のステップＳ２４で、受信されたコンテナ構造更新情報Ｍｓｇ．１に基づき、受信側サーバ１６上のコンテナエントリ階層の内容が変更される。

【００８９】

次に、図１４のフローチャートを用いてリーフエントリの同期管理方法について説明する。先ず、ステップＳ３０で、送信側クライアント１０によって、送信側サーバ１１で管理されているディレクトリ構造の、あるコンテナエントリの配下のリーフエントリが変更される。例えば、あるコンテナエントリの配下の新たなリーフエントリの追加や、あるコンテナエントリの配下のリーフエントリの削除や修正といった処理が行われる。

【００９０】

次のステップＳ３１では、送信側サーバ１１のあるコンテナエントリの配下のリーフエントリに対してなされた変更が送信側レプリケータ１２によって検知される。そして、検知結果に基づき、あるコンテナエントリ配下のリーフエントリの変更によるリーフ更新情報Ｍｓｇ．ｘ１が生成される。生成されたリーフ更新情報Ｍｓｇ．ｘ１は、放送ネットワーク２を介して複数の受信側レプリケータ１７に対してサイクリックに放送される。

【００９１】

放送されたリーフ更新情報Ｍｓｇ．ｘ１は、ステップＳ３２で、受信側レプリケータ１７によって受信される。受信側レプリケータ１７は、受信したリーフ更新情報Ｍｓｇ．ｘ１に基づき、受信側サーバ１６に格納されたディレクトリ情報に管理される、対応するリーフエントリを変更する。これにより、送信側１と受信側３とで、ディレクトリ情報のリーフエントリの同期がとられる。

【００９２】

リーフ更新情報Ｍｓｇ．ｘ１のフォーマットは、例えば、

Leaf Entry Update Message {

MessageID,

差分更新情報

}

このように定義される。「MessageID」（メッセージＩＤ）は、このメッセージ（リーフ更新情報Ｍｓｇ．ｘ１）の識別情報であって、例えば、このメッセージが生成される毎に１ずつ増加される整数である。また、「差分更新情報」は、上述したディレクトリ構造の差分更新情報である。

【００９３】

図１４のフローチャートにおけるステップＳ３１の処理を、図１５のフローチャートを用いて、より詳細に説明する。この図１５のフローチャートによる処理は、全て送信側レプリケータ１２上で行われる。先ず、ステップＳ４０で、送信側サーバ１１上の、あるコンテナエントリの配下のリーフエントリ名が全て読み込まれる。読み込まれたリーフエントリ名は、送信側レプリケータ１２が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー４として記憶される。

【００９４】

コピー１が記憶されたら、次のステップＳ４１で、タイマが所定の時間にセットされ、起動される。ステップＳ４２によって、タイマにセットされた所定時間を超過したかどうかが判断され、所定時間を超過したと判断されたなら、処理はステップＳ４３に移行する。ステップＳ４３では、再び送信側サーバ１１上の、あるコンテナエントリの配下のリーフエントリ名が全て読み込まれる。読み込まれたリーフエントリ名は、送信側レプリケータ１２が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー５として記憶される。

【００９５】

次のステップＳ４４では、ステップＳ４０で記憶されたコピー４と、ステップＳ４３で記憶されたコピー５とが比較される。比較の結果、両者の間に差分が無いとされれば（ステップＳ４５）、処理はステップＳ４１へ戻され、再びタイマがセットされ、コピー５の記憶が行われる。

【００９６】

一方、ステップＳ４４で、コピー４とコピー５との間に差分があるとされれば、処理はステップＳ４６に移行する。ステップＳ４６では、コピー４とコピー５との差分に基づき、差分更新情報が生成される。そして、この差分更新情報が記述されたリーフ更新情報Ｍｓｇ．ｘ１が生成される。生成されたリーフ更新情報Ｍｓｇ．ｘ１は、放送ネットワーク２に対して送信され、放送される。放送されたリーフ更新情報Ｍｓｇ．ｘ１は、複数の受信側レプリケータ１７に受信される。

【００９７】

ステップＳ４６でリーフ更新情報Ｍｓｇ．ｘ１が放送されると、次のステップＳ４７で、コピー４の内容がコピー５の内容で書き替えられ、処理はステップＳ４１に戻される。

【００９８】

なお、上述の図１５のフローチャートの処理は、送信側レプリケータ１２によって、送信側サーバ１１が管理するディレクトリ構造上の全てのコンテナエントリに対して行われる。

【００９９】

図１４のフローチャートにおけるステップＳ３２の処理を、図１６のフローチャートを用いて、より詳細に説明する。この図１６のフローチャートによる処理は、全て受信側レプリケータ１７上で行われる。最初のステップＳ５０で、送信側レプリケータ１２によって放送ネットワーク２を介して放送されたリーフ更新情報Ｍｓｇ．ｘ１が、受信側レプリケータ１７によって受信される。

【０１００】

ステップＳ５１で、ステップＳ５０での受信がリーフ更新情報Ｍｓｇ．ｘ１の初回の受信であるかどうかが判断される。若し、この受信が初回の受信で無いと判断されたら、処理はステップＳ５３に移行し、受信されたリーフ更新情報Ｍｓｇ．ｘ１のメッセージＩＤが受信側レプリケータ１７が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー６として記憶される。

【０１０１】

次のステップＳ５４では、受信されたリーフ更新情報Ｍｓｇ．ｘ１の内容、すなわち、リーフ更新情報Ｍｓｇ．ｘ１に記述された差分更新情報に基づき、受信側サーバ１６で管理されているディレクトリ情報のうち、対応するリーフエントリの内容が変更される。ステップＳ５４の処理の後、処理はステップＳ５０に戻される。

【０１０２】

一方、上述のステップＳ５１で、ステップＳ５０でのリーフ更新情報Ｍｓｇ．ｘ１の受信が初回の受信では無いと判断されたら、処理はステップＳ５２に移行する。ステップＳ５２では、受信されたリーフ更新情報Ｍｓｇ．ｘ１に記述されるメッセージＩＤと、前回の受信の際のステップＳ５３の処理でコピー６として記憶されたメッセージＩＤとが同一であるかどうかが判断される。若し、同一であるとされれば、処理はステップＳ５０に戻される。

【０１０３】

一方、ステップＳ５２で、両者のメッセージＩＤが異なるとされれば、処理はステップＳ５３に移行する。ステップＳ５３では、上述したように、メッセージＩＤが記憶媒体にコピー６として記憶される。この場合には、新たに受信されたメッセージＩＤで、前回受信され記憶されたメッセージＩＤが例えば上書きされることになる。そして、次のステップＳ５４で、受信されたリーフ更新情報Ｍｓｇ．ｘ１に基づき、受信側サーバ１６上の対応するリーフエントリの内容が変更される。

【０１０４】

ところで、上述したように、図１４のフローチャート中のステップＳ３１による、リーフ更新情報Ｍｓｇ．ｘ１の放送は、送信側１で管理されるディレクトリ構造中の、全コンテナエントリのそれぞれに関して行われ、放送されるリーフ更新情報Ｍｓｇ．ｘ１は、膨大な量になることが予想される。したがって、従来技術で問題点として既に述べたように、受信側３において全てのリーフ更新情報Ｍｓｇ．ｘ１を受信し、図１６のフローチャートによる処理を行うことは、大きな負担になる。そのため、受信側３では、必要とされている、すなわち、頻繁に照会されるコンテナエントリの配下のリーフエントリに対するリーフ更新情報Ｍｓ

ｇ．ｘ１のみを、放送された多数のリーフ更新情報Ｍｓｇ．ｘ１から、効率よくフィルタ処理する必要がある。

【０１０５】

例えば、受信側レプリケータ１７が家庭内でテレビジョン受像機などに接続して用いられる、セットトップボックス（ＳＴＢ）のような、処理能力や記憶容量が限定された、すなわち、コンピュータリソースに制約のある環境に実装される場合を想定する。この場合には、放送されるリーフ更新情報Ｍｓｇ．ｘ１の取得には限度がある。そのため、受信されたリーフ更新情報Ｍｓｇ．ｘ１を取捨選択して装置内に取り込み、記憶コストやメッセージ処理コストの軽減を図る必要がある。すなわち、受信側レプリケータ１７において、不必要な記憶や処理にかかるコストを制限する必要がある。特に、ディレクトリサービスが普及し、送信側サーバ１１で管理される対象のディレクトリ構造が巨大になるのに伴い、上述のリーフ更新情報Ｍｓｇ．ｘ１の取捨選択は、より重要な事項となってくる。

【０１０６】

以下に、リーフ更新情報Ｍｓｇ．ｘ１に対してフィルタ処理を行う方法について説明し、さらに、この発明の主旨に係る、フィルタ処理を効率的に行う方法について説明する。送信側レプリケータ１２は、放送されるリーフ更新情報Ｍｓｇ．ｘ１に対して、受信側レプリケータ１７においてフィルタ処理を行うためのフィルタリングマスクを付加する。フィルタリングマスクを解釈するためのマスクスキーマ構造と、マスクスキーマ構造を送信側レプリケータ１２から受信側レプリケータ１７に通知する方法などについては、後述する。

【０１０７】

リーフ更新情報Ｍｓｇ．ｘ１にフィルタリングマスクを付加したメッセージ（Ｍｓｇ．ｘ１’）の構造を、次のように定義し、上述のリーフ更新情報Ｍｓｇ．ｘ１を全てこのリーフ更新情報Ｍｓｇ．ｘ１’で置き替える。すなわち、リーフ更新情報Ｍｓｇ．ｘ１’は、

```
Leaf Entry Update Message  {
    MessageID,
    FilteringMask,
```

差分更新情報

}

このように定義される。「MessageID 」（メッセージＩＤ）は、上述のリーフ更新情報Ｍｓｇ．ｘ１の場合と同様に、このメッセージ（リーフ更新情報Ｍｓｇ．ｘ１'）の識別情報であって、例えば、このメッセージが生成される毎に１ずつ増加される整数である。「差分更新情報」は、ここに記述されるフィルタリングマスクで特定されるコンテナエントリ配下のリーフエントリの、追加、削除および属性変更といった手続の情報である。

【０１０８】

「FilteringMask 」（フィルタリングマスク）の構造は、次のように定義される。すなわち、フィルタリングマスクは、

FilteringMask {

MaskSchema Version,

Mask Value

}

このように定義される。「MaskSchema Version」（マスクスキーマバージョン）は、例えば上述のコンテナ構造更新情報Ｍｓｇ．１におけるメッセージＩＤに相当し、例えばこのフィルタリングマスクが生成される毎に１、増加される値である。「Mask Value」（マスク値）は、例えばビット列あるいはバイト単位で表されるマスクの値である。

【０１０９】

なお、マスク値の構造は、マスクスキーマバージョンによって対応付けられるマスクスキーマ（後述する）によって規定される。マスクスキーマは、後述する別のメッセージによって送信側レプリケータ１２から受信側レプリケータ１７に通知される。

【０１１０】

マスク値の割当方法について説明する。この実施の一形態では、あるコンテナエントリの配下のコンテナエントリのそれぞれを、所定ビット数からなるビット列で識別する。受信側レプリケータ１７では、受信されたリーフ更新情報Ｍｓｇ

．ｘ１’中に記述されるマスク値を参照することによってフィルタ処理を行い、必要なリーフ更新情報Ｍｓｇ．ｘ１’を選択的に抽出することができる。

【０１１１】

フィルタリングマスクのマスク値のビット配列構造は、コンテナエントリの階層構造に対応させて決定される。例えば図１７Ａに一例が示されるように、図３で説明したエントリ名の記述方法に倣い、上位のコンテナエントリＸの配下のエントリＸ．Ａ、Ｘ．Ｂ、Ｘ．Ｃ、Ｘ．ＤおよびＸ．Ｅを互いに識別するために、それぞれ３ビットのマスク値（０００）、（００１）、（０１０）、（０１１）および（１００）が割り当てられる。なお、〔...〕は、より上位のコンテナエントリが存在することを示す。

【０１１２】

このようにマスク値が与えられた、コンテナエントリＸの配下のコンテナエントリに対して、エントリの追加や削除が行われた場合、図１８のフローチャートに従って処理が行われ、コンテナエントリの増減に応じてマスク値の割り当てを行う。なお、以下では、コンテナエントリの追加や削除が行われる前のコンテナ階層を、更新前コンテナ階層と称する。更新前コンテナ階層のマスク桁数Ｍ’は、送信側レプリケータ１２の例えばメモリに記憶されているものとする。

【０１１３】

先ず、最初のステップＳ６０で、送信側レプリケータ１２によって、対象となるコンテナエントリの配下のコンテナエントリの数Ｎが取得される。コンテナエントリ中の、配下のコンテナエントリのリストを参照することで、コンテナエントリ数Ｎが求められる。次のステップＳ６１では、Ｎ個の要素を一意に識別可能なビット数Ｍが選ばれ、マスクの桁数がＭ桁とされる。例えば、上述した図１７Ａの例では、コンテナエントリＸは、配下のコンテナエントリを５個有しているので、５個を一意に識別可能なビット数〔３〕がマスクの桁数とされる。

【０１１４】

次に、ステップＳ６２で、上述のステップＳ６１で割り当てられたビット数Ｍが、対応する更新前コンテナ階層に割り当てられたマスク桁数Ｍ’と同一かどうかが判断される。若し、マスク桁数Ｍとマスク桁数Ｍ’とが同一であれば、処理

はステップＳ６３に移行する。

【０１１５】

ステップＳ６３では、更新後のコンテナ階層のコンテナエントリのうち、更新前コンテナ階層のコンテナエントリに対応するエントリには、同一のマスク値を割り当てる。さらに、次のステップＳ６４で、例えば更新後のコンテナ階層に新規にコンテナエントリの追加が起こったなどして、更新後のコンテナ階層に、更新前コンテナ階層に対応するコンテナエントリが存在しないコンテナエントリがあった場合、そのコンテナエントリに対して、同一のコンテナ階層の他のコンテナエントリのマスク値と重複しないようなマスク値が与えられる。

【０１１６】

一方、上述のステップＳ６２で、マスク桁数Ｍとマスク桁数Ｍ’とが同一ではないと判断されたら、処理はステップＳ６５に移行し、当該コンテナ階層の全コンテナエントリのそれぞれに対して、一意にマスク値が与えられる。

【０１１７】

例えば、上述の図１７Ａの状態に、新たにコンテナエントリ”...Ｘ．Ｆ”が追加され、図１７Ｂのようなコンテナ階層になったとする。コンテナエントリ”...Ｘ”の配下のコンテナエントリ数Ｎは、６であり、一意に表現するためには３ビットが必要とされ、更新後のコンテナエントリ”...Ｘ”の配下のコンテナ階層のマスク桁数Ｍ＝３でる。更新前コンテナ階層のマスク桁数Ｍ’＝３であって、マスク桁数Ｍ’とマスク桁数Ｍとは等しい。したがって、図１７Ｂに示される各コンテナエントリ”...Ｘ．Ａ”、”...Ｘ．Ｂ”、”...Ｘ．Ｃ”、”...Ｘ．Ｄ”および”...Ｘ．Ｅ”は、更新前コンテナ階層の対応するエントリのマスク値がそれぞれ割り当てられる（ステップＳ６３）。一方、新規に追加されたコンテナエントリ”...Ｘ．Ｆ”は、同じコンテナ階層の他のコンテナエントリと重複しないように、マスク値（１０１）が割り当てられる（ステップＳ６４）。

【０１１８】

また例えば、上述の図１７Ａの状態から、コンテナエントリ”...Ｘ．Ｃ”を削除して、図１７Ｃのようなコンテナ階層になったとする。この場合、コンテナ

エントリ”... X”配下のコンテナエントリ数Nは、４であり、２ビットのマスク桁数Mで各コンテナエントリを識別可能なので、更新後のコンテナ階層のマスク桁数M＝２である。一方、更新前コンテナ階層のマスク桁数M’＝３であって、更新前と更新後とで、マスク桁数が異なる。したがって、ステップＳ６５の処理によって、当該階層の全エントリに、マスク桁数M＝２で新たにマスク値が割り当てられる。

【０１１９】

さらに、図１７Ｃの状態に、新たにコンテナエントリ”... X．G”が追加され、図１７Ｄの状態になったとする。この場合、コンテナエントリ”... X”配下のコンテナエントリ数Nは５であり、コンテナエントリを互いに識別するためには、マスク桁数M＝３とする必要がある。マスク桁数Mが更新前のマスク桁数M’＝２と異なるため、ステップＳ６５の処理によって、コンテナエントリ”... X”の配下の全コンテナエントリに、新たにマスク値が割り当てられる。

【０１２０】

マスク値のビットアサインは、ディレクトリ構造の上位側から、コンテナ階層順にシリアルになされる。一方、この実施の一形態では、上述のように、同一階層に存在するエントリ数によってマスク桁数が異なる。また、エントリの削除や追加などによって、コンテナ階層中のエントリ数が変化し、それに伴いマスク桁数が変化する。そのため、マスク値を表すビット列中のどのビットがどのコンテナエントリ（あるいはコンテナ階層）に対応するかを判断し、マスク値を解釈するための情報機構が必要となる。

【０１２１】

この実施の一形態では、マスク値を解釈するための情報機構として、次に示すマスクスキーマ(MaskSchema)を定義する。マスクスキーマは、

```
MaskSchema {
    MaskSchema Version,
    TotalMaskLength,
    Set of ContainerEntryMaskSchema
}
```

このように定義される。「MaskSchema Version」（マスクスキーマバージョン）は、例えば上述のコンテナ構造更新情報Ｍｓｇ．１におけるメッセージＩＤに相当し、例えば対応するフィルタリングマスクが生成される毎に１、増加される値である。「TotalMaskLength 」（全マスク長）は、全体のコンテナ階層に対応する、マスク値全体のビット長を表す。すなわち、全マスク長は、ディレクトリ構造の全ての階層を表現するために必要なビット数に対応する。「Set of ContainerEntryMaskSchema 」（セットオブコンテナエントリマスクスキーマ）は、後述する「 ContainerEntryMaskSchema 」（コンテナエントリマスクスキーマ）の配列を表す。

【０１２２】

上述のコンテナエントリマスクスキーマは、あるコンテナエントリに対応するフィルタリングマスクを規定する。すなわち、コンテナエントリマスクスキーマは、

```
ContainerEntryMaskSchema {
    ContainerEntryName,
    OffsetLength,
    MaskLength,
    AssignedMaskValue
}
```

このように定義される。「ContainerEntryName」（コンテナエントリ名）は、対象となるコンテナエントリのエントリ名を表す文字列である。「OffsetLength」（オフセット長）は、このコンテナエントリに対応するフィルタリングマスクの、全マスク値の最初のビットからのオフセット値であり、「MaskLength」（マスク長）は、マスク値の桁数（ビット長）である。「AssignedMaskValue 」（割り当てマスク値）は、対象となるコンテナエントリに割り当てられたマスク値であり、ビット列で表される。

【０１２３】

図１９を用いて、コンテナエントリマスクスキーマの符号化について説明する。図１９Ａは、上述の図１７Ａに対応する図であって、上位のコンテナエントリ

”...X”の配下に、コンテナエントリ名”...X.A”、”...X.B”、”...X.C”、”...X.D”および”...X.E”の５つのコンテナエントリが存在し、それぞれ３桁のマスク長で以てマスク値が割り当てられている。なお、ここでは説明のため、これら５つのコンテナエントリは、配下に他のエントリを有しないものとする。

【０１２４】

図１９Ｂは、コンテナエントリ”...X.C”のマスク値の一例を示す。この例では、オフセット長が７７ビットであることから、コンテナエントリ”...X.C”に３ビットのマスク長で割り当てられた割り当てマスク値が、コンテナエントリ”...X.C”のマスク値の７８ビット目から開始される３ビットであることが分かる。オフセット長に含まれる７７ビットのマスク値は、コンテナエントリ”...X.C”より上位のコンテナエントリに対応する割り当てマスク値である。

【０１２５】

このように、マスク値における対象コンテナエントリの割り当てマスク値の位置が規定され、コンテナエントリマスクスキーマが符号化される。

【０１２６】

コンテナエントリマスクスキーマのより具体的な例を示す。上述したコンテナエントリ”...X.C”に対応するコンテナエントリマスクスキーマは、例えば、

```
ContainerEntryMaskSchema {
    "...X.C",   (ContainerEntryName)
    77,         (OffsetLength)
    3,          (MaskLength)
    010         (AssignedMaskValue)
}
```

このようになる。なお、括弧()内は、説明のためのものであって、実際に記述する必要は無い。

【０１２７】

また、図１９Ａに示されるコンテナエントリ”... X. Dに対応するコンテナエントリマスクスキーマは、例えば、

```
ContainerEntryMaskSchema {
    "...X.D",
    77,
    3,
    011
}
```

このようになる。

【０１２８】

このときのマスクスキーマは、例えばマスクスキーマバージョンを４９８、全マスク長を１３４ビットとした場合、

```
MaskSchema {
    498,   (MaskSchema Version)
    134,   (TotalMaskLength)
....
    ContainerEntryMaskSchema {
        "...X.C",
        77,
        3,
        010
    }
    ContainerEntryMaskSchema {
        "...X.D",
        77,
        3,
        011
    }
....
```

}

このようになる。上述の例では、コンテナエントリ”... X．Cおよび”... X．Dのコンテナエントリマスクスキーマがマスクスキーマ中に記述されているが、〔....〕の部分には、さらに他のコンテナエントリマスクスキーマが記述される。この例で分かるように、マスクスキーマには、一つのディレクトリ構造における全コンテナエントリに関するコンテナエントリマスクスキーマが記述される。

【０１２９】

なお、この例で、全マスク長が１３４ビットとなっているのに対して、コンテナエントリ”... X．Cおよび”... X．Dについてのコンテナエントリマスクスキーマでは、オフセット値が７７ビットおよびマスク長が３ビットの、合計で８０ビットである。これは、これらコンテナエントリ”... X．Cおよび”... X．Dの配下にも、さらにコンテナ階層が存在することを示している。

【０１３０】

上述したマスクスキーマにおいて、コンテナエントリ”... X．Cに対応するフィルタリングマスクの符号化は、例えば、

```
FilteringMask {
    498,    (MaskSchema Version)
    .....................010    (Mask Value)
}
```

このようになる。なお、マスク値(Mask Value)は、〔０１１〕以外の部分も全て、他の階層のコンテナエントリの割り当てマスク値からなるビットで埋められる。

【０１３１】

同様に、コンテナエントリ”... X．Dに対応するフィルタリングマスクの符号化は、例えば、

```
FilteringMask {
    498,    (MaskSchema Version)
    .....................011    (Mask Value)
```

}

このようになる。

【０１３２】

送信側レプリケータ１２では、送信側サーバ１１をモニタして、コンテナエントリの階層構造の変更を検知して、上述したマスクスキーマの変更を行う。したがって、受信側３において適切なフィルタ処理を行うためには、送信側レプリケータ１２によって、階層構造の変更に基づく差分更新情報の通知と共に、変更されたマスクスキーマが受信側レプリケータ１７に通知される必要がある。

【０１３３】

この発明では、マスクスキーマを送信側レプリケータ１２から受信側レプリケータ１７に通知するために、上述したコンテナ構造更新情報Ｍｓｇ．１の構造に対して、マスクスキーマ構造を追加する。マスクスキーマ構造を追加されたコンテナ構造更新情報Ｍｓｇ．１’を、

```
Container Structure Update Message {
    MessageID,
    差分更新情報,
    MaskSchema
}
```

このように定義する。マスクスキーマは、コンテナ階層の構成が変更される毎に変更される可能性がある。そのため、このコンテナ構造更新情報Ｍｓｇ．１’も、コンテナ階層の構成の変更に応じて生成される。「MessageID 」（メッセージＩＤ）は、コンテナ構造更新情報Ｍｓｇ．１’が生成される毎に１ずつ増加される整数である。以下では、上述したコンテナ構造更新情報Ｍｓｇ．１を、全てこのコンテナ構造更新情報Ｍｓｇ．１’に置き替えるものとする。

【０１３４】

送信側レプリケータ１２では、上述した図１５のフローチャートにおけるステップＳ４６で、コンテナ階層に対応させたフィルタリングマスクが付加されたメッセージである、リーフ更新情報Ｍｓｇ．ｘ１’を生成し、受信側レプリケータ１７に放送している。ここで、受信側３では、リーフ更新情報Ｍｓｇ．ｘ１’に

よる受信側レプリケータ１７でのフィルタ処理を行う前に、受信側クライアント１５が必要としているコンテナ階層の対象部分を特定しておく必要がある。

【０１３５】

この実施の一形態では、受信側レプリケータ１７において、対象となるコンテナ階層をフィルタ処理するためのマスクをリストにした、ターゲットマスクリストを作成する。

【０１３６】

図２０を用いてターゲットマスクリストについて説明する。先ず、図２０Ａに示されるようなディレクトリ構造を想定する。図２０Ａのディレクトリ階層は、最上位のルートエントリ以外は全てコンテナエントリで構成されているものとする。各四角はコンテナエントリを表し、二重線の四角で示されるコンテナエントリは、例えばユーザの嗜好に基づき受信側クライアント１５でフィルタ処理を行うように特定されたエントリである。各エントリ内に表示された数字は、当該エントリ毎に割り当てられたマスク値である。

【０１３７】

図２０Ａに示されるように、ユーザの嗜好に基づきフィルタ処理するように受信側クライアント１５で特定されたコンテナエントリのそれぞれに対して、マスク１～５が割り当てられている。このディレクトリ構造において、マスク１～５の全マスク長分のマスク値は、ディレクトリ構造を上位側から辿り、マスク１が〔０００〕、マスク２が〔００１０〕マスク３が〔０１０〕、マスク４が〔１０００〕およびマスク５が〔１００１０〕となる。

【０１３８】

図２０Ｂは、このように特定されたマスクがリストとされたターゲットマスクリストの一例を示す。ターゲットマスクリストは、ディレクトリの構造を特定するスキーマバージョンと、ユーザの嗜好などに基づき受信側クライアント１５で特定された上述したマスク値のリストとからなる。すなわち、このターゲットマスクリストは、スキーマバージョンで記述されたディレクトリ構造でのみ有効なリストである。

【０１３９】

図２１は、ターゲットマスクリストを作成する処理のフローチャートである。このフローチャートは、受信側レプリケータ１７で実行される。先ず、最初のステップＳ７０で、受信側レプリケータ１７によって、コンテナ構造更新情報Ｍｓｇ．１’が受信される。ステップＳ７１で、ステップＳ７０での受信がコンテナ構造更新情報Ｍｓｇ．１’の初回の受信であるかどうかが判断され、初回の受信であると判断されれば、処理はステップＳ７３に移行する。

【０１４０】

ステップＳ７３では、受信されたコンテナ構造更新情報Ｍｓｇ．１’のメッセージＩＤを、受信側レプリケータ１７が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー７として記憶される。

【０１４１】

次のステップＳ７４で、受信されたコンテナ構造更新情報Ｍｓｇ．１’の内容に基づき、コンテナ階層を生成する。生成されたコンテナ階層を示す情報が受信側レプリケータ１７から受信側クライアント１５に提示され、特定すべきコンテナエントリの選択が促される。例えば、受信側クライアント１５では、所定の表示手段を用いて、供給されたコンテナ階層を示す情報に基づく表示を行う。ユーザは、この表示に基づき、所定の方法で必要なコンテナエントリを選択する。選択されたコンテナエントリの情報は、受信側クライアント１５から受信側レプリケータ１７に渡される。

【０１４２】

なお、コンテナエントリの特定は、ユーザの直接的な選択によってなされるのに限定されない。例えば、受信側クライアント１５によって、ユーザが照会を行ったコンテナエントリの情報を蓄積し、蓄積された情報に基づきユーザの嗜好の傾向を学習して自動的に必要と思われるコンテナエントリを選択するようにもできる。さらに、ユーザによる直接的な選択と、学習による自動的な選択とを併用することもできる。

【０１４３】

このようにして、ステップＳ７４でコンテナエントリが選択されると、ステップＳ７５で、選択されたコンテナ階層に対応するフィルタリングマスクが設定さ

れる。設定されたフィルタリングマスクの一覧は、ターゲットマスクリストとして、例えば受信側レプリケータ１７が有する、例えばメモリやハードディスクといった記録または記憶媒体に記憶される。

【０１４４】

一方、上述のステップＳ７１で、コンテナ構造更新情報Ｍｓｇ．１’の受信が初回ではないと判断されれば、処理はステップＳ７２に移行する。ステップＳ７２では、受信されたコンテナ構造更新情報Ｍｓｇ．１’のメッセージＩＤが、前回までのコンテナ構造更新情報Ｍｓｇ．１’の受信によりステップＳ７３でコピー７として記憶媒体に記憶されたメッセージＩＤと同一かどうかが判断される。

【０１４５】

若し、両者が同一であると判断されたら、処理はステップＳ７０に戻される。一方、ステップＳ７３で両者が同一では無いと判断されたら、処理はステップＳ７４に移行し、今回受信されたコンテナ構造更新情報Ｍｓｇ．１’のメッセージＩＤが前回までのメッセージＩＤの代わりに記憶媒体に記憶され、今回受信されたコンテナ構造更新情報Ｍｓｇ．１’に基づき以降の処理が行われる。

【０１４６】

図２２は、図２１のフローチャートに従って作成されたターゲットマスクリストに基づき、放送されたリーフ更新情報Ｍｓｇ．ｘ１’を選択的に受信する処理を示すフローチャートである。受信側レプリケータ１７は、ターゲットマスクリストに列挙されたフィルタリングマスクを有するリーフ更新情報Ｍｓｇ．ｘ１’を、放送ネットワーク２で放送されたリーフ更新情報Ｍｓｇ．ｘ１’の中から選択的に受信する。選択的に受信されたリーフ更新情報Ｍｓｇ．ｘ１’により、上述した図１４のフローチャートにおけるステップＳ３２の処理が実行される。

【０１４７】

図２２において、先ず、最初のステップＳ８０で、受信側レプリケータ１７によって、放送ネットワーク２によって放送されたリーフ更新情報Ｍｓｇ．ｘ１’が受信される。受信側レプリケータ１７では、記憶媒体に記憶されているターゲットマスクリストが参照され、受信されたリーフ更新情報Ｍｓｇ．ｘ１’に示されるフィルタリングマスクがターゲットマスクリストに存在するかどうかが判断

される。若し、ターゲットマスクリスト中に存在しなければ、処理はステップＳ８０に戻される。

【０１４８】

一方、ステップＳ８１で、当該フィルタリングマスクがターゲットマスクリスト中に存在すると判断されれば、処理はステップＳ８２に移行し、ステップＳ８０での受信がリーフ更新情報Ｍｓｇ．ｘ１’の初回の受信であるかどうかが判断される。若し、初回の受信であるとされれば、処理ステップＳ８４へ移行し、受信されたリーフ更新情報Ｍｓｇ．ｘ１’に示されるメッセージＩＤが、例えば受信側レプリケータ１７が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー８として記憶される。そして、次のステップＳ８５で、受信されたリーフ更新情報Ｍｓｇ．ｘ１’が受信側レプリケータ１７での処理の対象として選択される。

【０１４９】

一方、ステップＳ８２で、上述のステップＳ８０でのリーフ更新情報Ｍｓｇ．ｘ１’の受信が初回の受信では無いと判断されたら、処理はステップＳ８３に移行する。ステップＳ８３では、受信されたリーフ更新情報Ｍｓｇ．ｘ１’のメッセージＩＤが、前回までのリーフ更新情報Ｍｓｇ．ｘ１’の受信によりステップＳ８４でコピー８として記憶媒体に記憶されたメッセージＩＤと同一かどうかが判断される。

【０１５０】

若し、両者が同一であると判断されたら、処理はステップＳ８０に戻される。一方、ステップＳ８３で両者が同一では無いと判断されたら、処理はステップＳ８４に移行し、今回受信されたコンテナ構造更新情報Ｍｓｇ．ｘ１’のメッセージＩＤが前回までのメッセージＩＤの代わりに記憶媒体に記憶され、今回受信されたコンテナ構造更新情報Ｍｓｇ．ｘ１’に基づき以降の処理が行われる。

【０１５１】

上述のようにして、受信側サーバ１６で管理されるディレクトリ構造には、送信側サーバ１１で管理されるディレクトリ構造のうち、受信側サーバ１６を利用するユーザの嗜好を反映した部分のみを蓄積し、更新することができる。そのた

め、受信側サーバ１６においてディレクトリ構造を蓄積するための蓄積記憶媒体を有効に利用することができる。また、それと共に、受信側サーバ１６でのディレクトリ構造の格納コストを抑えることができる。さらに、受信側クライアント１５による受信側サーバ１６の内容の検索要求に対する処理効率を大幅に向上させることが可能となる。

【０１５２】

なお、上述では、各コンテナエントリに割り当てられた割り当てマスク長を可変長としたが、これはこの例に限定されない。割り当てマスク長は、例えばバイト単位などの固定長とすることもできる。

【０１５３】

次に、この発明の実施の一形態について説明する。この発明では、上述したディレクトリの放送型の差分更新方式を、実際の公開鍵証明書ディレクトリの放送型差分更新方式に適用する。先ず、図２３を用いて、公開鍵証明書ディレクトリについて説明する。図２３は、ユーザ（加入者）と認証局とからなる階層構造である、認証局構造の一例を概略的に示す。

【０１５４】

図２３において、ＣＡ＿０、ＣＡ＿１およびＣＡ＿２が認証局、ＥＥ＿１、ＥＥ＿２およびＥＥ＿３がエンドエンティティ（加入者）であるとする。実線矢印は、矢印の元の認証局が矢印の先の認証局若しくはエンドエンティティの公開鍵の証明書を発行することを意味している。

【０１５５】

図２３の例では、認証局ＣＡ＿１およびＣＡ＿２は、それぞれ認証局ＣＡ＿０により認証され、公開鍵証明書を発行される。エンドエンティティＥＥ＿１およびＥＥ＿２は、認証局ＣＡ＿１に認証され、公開鍵証明書を発行される。同様に、エンドエンティティＥＥ＿３は、認証局ＣＡ＿２により認証され、公開鍵証明書を発行される。一方、認証局ＣＡ＿０は、エンドエンティティＥＥ＿１、ＥＥ＿２およびＥＥ＿３を直接的には認証していない。このように、各認証局およびエンドエンティティの間で階層構造が形成される。

【０１５６】

また、点線矢印は、矢印の元のエンドエンティティが矢印の先の認証局と緊密な関係を結び、ルート認証局として信頼していることを意味する。すなわち、点線矢印の元のエンドエンティティは、矢印の先の認証局の公開鍵をルート公開鍵として利用する。図２３の例では、例えば、エンドエンティティＥＥ＿１は、認証局ＣＡ＿０と緊密な関係を結び、エンドエンティティＥＥ＿１は、ルート公開鍵としてＣＡ＿０の公開鍵を用いる。

【０１５７】

ここで、一例として、エンドエンティティＥＥ＿１がエンドエンティティＥＥ＿２の公開鍵を得る場合について考える。この場合、エンドエンティティＥＥ＿１は、次の２つの証明書の証明書パスを処理することになる。すなわち、第１の証明書パスは、認証局ＣＡ＿０によって発行された認証局ＣＡ＿１の証明書を得るためのパスである。第２の証明書パスは、認証局ＣＡ＿１によって発行されたエンドエンティティＥＥ＿２の証明書を得るためのパスである。

【０１５８】

つまり、エンドエンティティＥＥ＿１にとっては、エンドエンティティＥＥ＿２の認証を行っている認証局ＣＡ＿１が信頼できるかどうかが分からないため、自分と信頼関係にある認証局ＣＡ＿０に遡って認証局ＣＡ＿１の認証を確認する必要がある。

【０１５９】

また、他の例として、エンドエンティティＥＥ＿１がエンドエンティティＥＥ＿３の公開鍵を得る場合について考える。この場合、上述と同様に、エンドエンティティＥＥ＿１にとっては、エンドエンティティＥＥ＿３の認証を行っている認証局ＣＡ＿２が信頼できるかどうか分からないため、エンドエンティティＥＥ＿１は、次の２つの証明書の証明書パスを処理することになる。すなわち、第１の証明書パスは、認証局ＣＡ＿０によって発行された認証局ＣＡ＿２の証明書を得るためのパスであり、第２のパスは、認証局ＣＡ＿２によって発行されたエンドエンティティＥＥ＿３の証明書を得るためのパスである。

【０１６０】

さらに他の例として、エンドエンティティＥＥ＿２がエンドエンティティＥＥ

＿１の公開鍵を得る場合について考える。この場合には、エンドエンティティＥＥ＿１およびＥＥ＿２は、共通の認証局ＣＡ＿１によって認証されている。そのため、エンドエンティティＥＥ＿２は、認証局ＣＡ＿１によって発行されたエンドエンティティＥＥ＿１の証明書を得るだけで事足りる。

【０１６１】

図２３からも分かるように、認証局構造は、送信側ディレクトリサーバ１１で管理されているディレクトリツリーに対応付けることが可能である。図２４は、認証局構造とディレクトリツリーとの一例の対応関係を示す。図２４Ａに示される認証局構造の各階層が、図２４Ｂに示されるディレクトリツリーの各階層にそれぞれ対応付けられる。すなわち、認証局構造における認証局ＣＡ＿０、ＣＡ＿１、ＣＡ＿２がディレクトリツリーにおけるコンテナエントリ１００、１０１および１０２にそれぞれ対応付けられる。認証局構造におけるエンドエンティティＥＥ＿１、ＥＥ＿２およびＥＥ＿３がリーフエントリ１１０、１１１および１１２にそれぞれ対応付けられる。

【０１６２】

ディレクトリツリーの各エントリに対するエントリ名も、認証局構造に対応して与えることができる。例えば、図３を用いて上述した命名法を倣えば、コンテナエントリ１００のエントリ名は、対応する認証局ＣＡ＿０に基づきエントリ名ＣＡ＿０とされる。コンテナエントリ１０１は、認証局構造において認証局ＣＡ＿０から階層構造を辿った認証局ＣＡ＿１に対応している。したがって、コンテナエントリ１０１には、エントリ名ＣＡ＿０．ＣＡ＿１が与えられる。同様に、リーフエントリ１１０は、認証局構造において、認証局ＣＡ＿１からさらに階層構造を辿ったエンドエンティティＥＥ＿１に対応している。したがって、リーフエントリ１１０には、エントリ名ＣＡ＿０．ＣＡ＿１．ＥＥ＿１が与えられる。他のエントリにも、同様にしてに消極構造に対応してエントリ名を与えることができる。

【０１６３】

認証局構造における認証局に対応するエントリ（以下、ＣＡエントリと称する）には、以下の２つの属性を持たせる。１つは、対応する認証局の最新の公開鍵

証明書のシリアル番号を格納する属性である。ここでは、この属性の属性名を"PublicKeyCertificateSerialNumber"とし、属性値をシリアル番号そのものとする。

【０１６４】

もう１つは、対応する認証局の最新の公開鍵証明書、若しくは、最新の公開鍵証明書を取得する取得方法を格納する属性である。ここでは、この属性の属性名を"LatestPublicKeyCertificate"とし、属性値を、最新の公開鍵証明書そのもの、若しくは、最新の公開鍵証明書の取得方法とする。最新の公開鍵証明書を取得する取得方法は、例えば、その公開鍵証明書が置かれているＵＲＬで示される。

【０１６５】

一方、認証局構造におけるエンドエンティティに対応するエントリ（以下、ＥＥエントリと称する）には、同様にして、以下の２つの属性を持たせる。１つは、対応するエンドエンティティの最新の公開鍵証明書のシリアル番号を格納する属性である。ここでは、この属性の属性名を"PublicKeyCertificateSerialNumber"とし、属性値を、シリアル番号そのものとする。

【０１６６】

もう１つは、対応するエンドエンティティの最新の公開鍵証明書、若しくは、最新の公開鍵証明書の取得方法を格納する属性である。ここでは、この属性の属性名を"LatestPublicKeyCertificate"とし、属性値を、最新の公開鍵証明書そのもの、若しくは、最新の公開鍵証明書の取得方法とする。最新の公開鍵証明書を取得する取得方法は、例えば、その公開鍵証明書が置かれているＵＲＬで示される。

【０１６７】

上述のように、この実施の一形態では、エントリの属性に対して、最新の公開鍵証明書を直接的に格納する場合と、最新の公開鍵証明書を取得するための情報を格納する場合との、２種類の形態が設けられている。このように、エントリ属性に対して２種類の情報を格納可能としておけば、放送ネットワーク資源を有効的に活用できる。

【０１６８】

一般に、公開鍵証明書は、その証明書を取得するための情報（例えばＵＲＬなど）に比べてデータサイズが非常に大きい。したがって、ディレクトリ構造が極めて多数のディレクトリエントリを有する場合には、それら多数のディレクトリエントリに対応する最新の公開鍵証明書を周期的に放送すると、帯域を大幅に圧迫してしまうことになる。しかしながら、帯域を節約するために、全て公開鍵証明書の取得情報のみの放送としてしまうと、受信側における最新の公開鍵の参照時に、通信ネットワークを介した公開鍵証明書の取得処理が必ず行われることになり、通信ネットワークのトラフィックを圧迫してしまう。

【０１６９】

そこで、放送ネットワークを介した即時広域配布のメリットを生かすために、この発明の実施の一形態では、以下のような方法を採る。すなわち、例えばとある公開鍵証明書が失効した場合、新たに発行された公開鍵証明書を、対応するエントリの属性"LatestPublicKeyCertificate"に格納すると共に、その公開鍵証明書のシリアル番号を"PublicKeyCertificateSerialNumber"に格納し、更新情報として放送する。所定の時間が経過したら、当該公開鍵証明書をとあるＵＲＬで指定されるアドレスに置き、エントリの属性"LatestPublicKeyCertificate"の内容を当該ＵＲＬに入れ替え、更新情報として放送する。こうすることで、帯域を有効活用することが可能となる。

【０１７０】

図２５は、この実施の一形態に適用可能な系の一例を示す。この図２５の例は、暗号化電子メールなどの暗号化通信に２つのエンドエンティティが関わる例である。また、この図では、認証局ＣＡ＿０およびＣＡ＿１が関わる。なお、図２５において、上述した図１と共通する部分には同一の番号を付し、詳細な説明を省略する。

【０１７１】

図２５において、受信側３’をなす受信側レプリケータ１７’、受信側サーバ１６’および受信側クライアント１５’は、受信側３をなす上述した受信側レプリケータ１７、受信側サーバ１６および受信側クライアント１５と同等なもので、放送ネットワーク２を介して送信側１の送信側サーバ１１で管理されるディレ

クトリツリーの複製を管理する。

【０１７２】

ここで、受信側クライアント１５および１５’が図２３および図２４で上述したエンドエンティティＥＥ＿１およびＥＥ＿２にそれぞれ対応するものとする。受信側クライアント１５および１５’は、それぞれ、例えばソフトウェアからなる暗号化通信モジュール５０および５０’を有する。受信側クライアント１５と１５’は、暗号化通信モジュール５０および５０’を用いて、通信ネットワーク５１を介して互いに暗号化通信を行う。

【０１７３】

さらに、送信側において、送信側クライアント１０および１０’が存在し、送信側クライアント１０’は、送信側クライアント１０と同様に、送信側サーバ１１で管理されるディレクトリツリーの内容を更新できる。これら送信側クライアント１０および１０’は、図２３および図２４で上述した、認証局階層における認証局ＣＡ＿０およびＣＡ＿１に対応する。

【０１７４】

なお、図２５では、受信側は、受信側３および３’の２つが示されているが、実際には、さらに多数の受信側構成が存在し、これら多数の受信側構成は、通信ネットワーク５１を介して互いに通信が可能とされている。また、送信側クライアントも、ここでは送信側クライアント１０および１０’の２つが示されているが、実際には、さらに多数が存在する。

【０１７５】

図２６は、受信側クライアント１５および１５’の間で行われる暗号化通信の一例の手順を示すフローチャートである。なお、図２６では、受信側レプリケータ１７および１７’をそれぞれ第１および第２の受信側レプリケータ、暗号化通信モジュール５０および５０’をそれぞれ第１および第２の暗号化通信モジュールと称している。

【０１７６】

先ず、このフローチャートの処理に先立ち、受信側クライアント１５および１５’は、それぞれ、自身の秘密鍵および公開鍵のペアを生成しているものとする

。また、受信側クライアント１５すなわちエンドエンティティＥＥ＿１は、送信側クライアント１０’すなわち認証局ＣＡ＿１から、自身の公開鍵について公開鍵証明書を発行してもらっているものとする。このとき、図２３を用いて上述した証明書パス、すなわち、認証局ＣＡ＿０が認証局ＣＡ＿１に証明書を発行し、認証局ＣＡ＿１がエンドエンティティＥＥ＿２の証明書を発行することを考慮に入れ、前提として、認証局ＣＡ＿１は、認証局ＣＡ＿０すなわち送信側クライアント１０から予め公開鍵証明書を発行してもらっているものとする。

【０１７７】

同様に、受信側クライアント１５’すなわちエンドエンティティＥＥ＿２は、送信側クライアント１０’すなわち認証局ＣＡ＿１から、自身の公開鍵について公開鍵証明書を発行してもらっているものとする。

【０１７８】

これらの、受信側クライアント１５および１５’各々の公開鍵についての公開鍵証明書のそれぞれは、受信側クライアント１５および１５’が共にアクセス可能な場所、例えばインターネット上のとあるＷｅｂサイトに格納するか、予め相互に交換し合うことにより、互いに必要なときに利用できるようにしておく。

【０１７９】

先ず、最初のステップＳ９０で、受信側クライアント１５において、暗号通信モジュール５０が利用され受信側クライアント１５の秘密鍵によってメッセージに署名される。次のステップＳ９１では、受信側クライアント１５により受信側クライアント１５’の公開鍵証明書が取得され、取得された公開鍵証明書に格納されている公開鍵が得られる。なお、ステップＳ９１の処理の前提として、受信側クライアント１５により送信側クライアント１０’の公開鍵証明書が取得され、そこに格納されている公開鍵が既に取得されている。次のステップＳ９２では、受信側クライアント１５において、暗号通信モジュール５０が用いられ、ステップＳ９１で得られた公開鍵によりステップＳ９０で署名されたメッセージが暗号化される。

【０１８０】

ステップＳ９２で暗号化されたメッセージは、ステップＳ９３で、通信ネット

ワーク５１を介して受信側クライアント１５’に送信される。受信側クライアント１５’では、受信側クライアント１５から送信されたメッセージを受信し、暗号通信モジュール５０’を用いて受信側クライアント１５’の秘密鍵で、受信されたメッセージを復号化する。復号化されたメッセージには、上述のステップＳ９０において、受信側クライアント１５の秘密鍵を用いて署名されている。

【０１８１】

次のステップＳ９５では、受信側クライアント１５’により、受信側クライアント１５の公開鍵証明書が取得され、取得された公開鍵証明書に格納されている公開鍵が得られる。そして、ステップＳ９６で、受信側クライアント１５’により、暗号化モジュール５０’が用いられ、ステップＳ９５で得られた公開鍵により上述のステップＳ９４で復号化されたメッセージの署名が確認される。

【０１８２】

上述したフローチャートのうち、ステップＳ９１およびステップＳ９５において公開鍵が取得される際に、受信側クライアント１５および１５’は、取得する公開鍵が失効していない有効な公開鍵証明書に基づくものであることを確認しなければならない。この確認は、これらの公開鍵証明書を発行した認証局に問い合わせることで可能である。しかしながら、一般的には、公開鍵証明書の発行数が多い認証局では、問い合わせの負荷が増大し、応答性能が極端に低下するという問題により、オンラインによる有効性の確認処理が不可能であることが多い。

【０１８３】

そこで、この発明の実施の一形態では、受信側クライアント１５や受信側クライアント１５’は、それぞれ利用する公開鍵の有効性について該当する認証局に問い合わせる代わりに、各自のローカルな環境に設置されている受信側サーバ１６や受信側サーバ１６’で管理されるディレクトリに問い合わせるようにする。受信側クライアント１５や１５’のローカルな環境に設置されている受信側サーバ１６や１６’は、例えば、家庭内ＬＡＮ(Local Area Network)上、マンション内などに設置される電話線集線装置、ケーブルテレビネットワークのヘッドエンドなどが想定できる。これにより、当該公開鍵証明書を発行した認証局への問い合わせの集中を分散化させることができる。

【０１８４】

上述した、認証局への問い合わせをローカルな環境に置き換えて行うには、以下の情報を更新する必要がある。第１に、受信側サーバ１６においては、受信側クライアント１５’すなわちエンドエンティティＥＥ＿２の公開鍵証明書が有効か否かを示す情報が更新されている必要がある。また、この例では、上述した証明書パスに従い、受信側クライアント１５すなわちエンドエンティティＥＥ＿１は、エンドエンティティＥＥ＿２に公開鍵証明書を発行する認証局ＣＡ＿１とは緊密な関係にない。したがって、受信側サーバ１６においては、送信側クライアント１０’、すなわち、エンドエンティティＥＥ＿２に対して公開鍵証明書を発行する認証局ＣＡ＿１の公開鍵証明書が有効であるか否かを示す情報も、最新の情報に更新されている必要がある。

【０１８５】

第２に、受信側サーバ１６’においては、受信側クライアント１５すなわちエンドエンティティＥＥ＿１の公開鍵証明書が有効であるか否かの情報が、最新の情報に更新されている必要がある。

【０１８６】

なお、公開鍵証明書が有効であるかどうか、すなわち、その公開鍵証明書が失効しているか否かは、当該公開鍵証明書と、当該公開鍵証明書に対応する最新の公開鍵証明書とを比較し、シリアル番号が一致しているかどうかを調べることで判断できる。この発明では、新たに発行された公開鍵証明書のシリアル番号を、エントリの属性"PublicKeyCertificateSerialNumber"に格納するようにしているため、このエントリ属性を調べることで、当該公開鍵証明書が有効であるかどうかを知ることができる。

【０１８７】

上述の図２４に示されるように、図２３に示される認証局構造と、受信側クライアント１５および１５’のディレクトリエントリとの対応付けがなされている。したがって、受信側サーバ１６で管理されるディレクトリにおいては、コンテナエントリ１０１（エントリ名ＣＡ＿０．ＣＡ＿１）およびリーフエントリ１１１（エントリ名ＣＡ＿０．ＣＡ＿１．ＥＥ＿２）の更新情報に注目する必要があ

る。同様に、受信側サーバ１６’で管理されるディレクトリにおいては、リーフエントリ１１０の更新情報を注目しなければいけない。

【０１８８】

受信側サーバ１５および１５’のそれぞれでは、上述の注目対象となるエントリの上位のコンテナエントリがフィルタリングマスクの対象コンテナエントリとして選択される。フィルタリングマスクの対象コンテナエントリの選択は、上述した図２１に従い、ステップＳ７４においてなされる。選択されたコンテナエントリに対応するフィルタリングマスクが、それぞれのターゲットマスクリストに格納されることになる。

【０１８９】

図２７を用いて、それぞれのターゲットマスクリストにどのようなフィルタリングマスクが格納されるかについて、より具体的に説明する。なお、図２７において、斜線が付された四角は、公開鍵証明書を取得する対象となるエントリを示す。また、点線で示された四角は、フィルタリングマスクに対応するコンテナエントリを示す。図２７Ａおよび図２７Ｂは、それぞれ受信側サーバ１６および１６’におけるフィルタリング対象のエントリを示す。

【０１９０】

受信側サーバ１６におけるフィルタリング対象のエントリは、図２７Ａに示されるように、エントリ名ＣＡ＿０．ＣＡ＿１のコンテナエントリ１０１と、エントリ名ＣＡ＿０．ＣＡ＿１．ＥＥ＿２のリーフエントリ１１１である。受信側レプリケータ１７によって、これらエントリ１０１および１１１をそれぞれ示すフィルタリングマスクがターゲットマスクリストに格納される。ターゲットマスクリストは、例えば受信側レプリケータ１７が有する記憶媒体に記憶される。

【０１９１】

同様にして、受信側サーバ１６’におけるフィルタリングフィルタリング対象のエントリは、図２７Ｂに示されるように、エントリ名ＣＡ＿０．ＣＡ＿１．ＥＥ＿１のリーフエントリ１１０である。受信側レプリケータ１７’によって、これらエントリ１１０を示すフィルタリングマスクがターゲットマスクリストに格納される。

【０１９２】

なお、送信側サーバ１１で管理されているディレクトリ情報と、受信側サーバ１６および１６’で管理されているディレクトリ情報とは、同期がとれている必要がある。この実施の一形態では、既に述べたように、所定の時間にセットされたタイマによって決められるタイミングで以て放送される、ディレクトリ情報の差分更新情報によって受信側サーバ１６および１６’で管理されているディレクトリ情報が更新されることで、送信側と受信側との同期が取られる。

【０１９３】

このとき、送信側と受信側の同期は、段階的にとるようにできる。例えば、上述のタイミングで差分更新情報による更新を行うと共に、より長い周期で全てのディレクトリ構造の放送を行い、受信側サーバ１６および１６’の更新を行うようにできる。さらに、受信側サーバ１６および１６’と送信側サーバ１１とを双方向通信が可能な通信回線で接続して、受信側サーバ１６および１６’側において、ターゲットマスクリストに格納されている情報については、随時、送信側サーバ１１に対して最新情報を要求し、送信側サーバ１１では、この要求に応じて、対応するディレクトリ構造の送信を行うようにできる。

【０１９４】

【発明の効果】

以上説明したように、この発明の実施の一形態では、ターゲットマスクリストを用いることによって、上述のようにして、受信側サーバ１６および１６’で管理されるディレクトリツリーの内容には、送信側サーバ１１で管理されるディレクトリツリー構成のうち、受信側クライアント１５および１５’がよくアクセスするエントリの情報だけを蓄積および更新することができる。これにより、受信側サーバ１６および１６’におけるディレクトリツリーの格納コストを抑えることができると共に、蓄積媒体の有効活用が図れる効果がある。

【０１９５】

また、この発明によれば、認証局およびエンドエンティティからなる階層構造である認証局構造と、送信側ディレクトリサーバおよび受信側ディレクトリサーバで管理されるディレクトリツリーとが対応付けられ、認証局構造において公開

鍵証明書を得るための証明書パスに対応してターゲットマスクリストが作成される。そのため、複数の受信側ディレクトリサーバの内容に対する複数の受信側ディレクトリサーバからの公開鍵証明書の失効情報検索用急に対する処理効率を、大幅に向上させることができる効果がある。

【図面の簡単な説明】

【図１】

この発明に適用できる系の一例を示す略線図である。

【図２】

複数の受信側が放送ネットワークに接続されることを説明するための略線図である。

【図３】

ディレクトリ構造を説明するための略線図である。

【図４】

コンテナエントリの構造の一例を示す略線図である。

【図５】

リーフエントリの構造の一例を示す略線図である。

【図６】

送信側レプリケータの機能を説明するための機能ブロック図である。

【図７】

受信側クライアントの機能を説明するための機能ブロック図である。

【図８】

受信側サーバの機能を説明するための機能ブロック図である。

【図９】

ディレクトリ構造の差分更新情報について説明するための略線図である。

【図１０】

ディレクトリ構造の差分更新情報について説明するための略線図である。

【図１１】

コンテナエントリの同期管理方法を説明するためのフローチャートである。

【図１２】

コンテナエントリの同期管理方法をより詳細に説明するためのフローチャートである。

【図１３】

コンテナエントリの同期管理方法をより詳細に説明するためのフローチャートである。

【図１４】

リーフエントリの同期管理方法を説明するためのフローチャートである。

【図１５】

リーフエントリの同期管理方法をより詳細に説明するためのフローチャートである。

【図１６】

リーフエントリの同期管理方法をより詳細に説明するためのフローチャートである。

【図１７】

フィルタリングマスクのマスク値のビット配列構造を説明するための略線図である。

【図１８】

エントリの追加や削除が行われた場合の、コンテナエントリの増減に応じたマスク値の割り当て処理のフローチャートである。

【図１９】

コンテナエントリマスクスキーマの符号化を説明するための略線図である。

【図２０】

ターゲットマスクリストを説明するための略線図である。

【図２１】

ターゲットマスクリストを作成する処理のフローチャートである。

【図２２】

ターゲットマスクリストに基づき、放送されたリーフ更新情報Ｍｓｇ．ｘ１'を選択的に受信する処理を示すフローチャートである。

【図２３】

認証局構造を説明するための図である。

【図２４】

認証局構造とディレクトリツリーとの一例の対応関係を示す略線図である。

【図２５】

実施の一形態に適用可能な系の一例を示す略線図である。

【図２６】

受信側クライアントの間で行われる暗号化通信の一例の手順を示すフローチャートである。

【図２７】

それぞれのターゲットマスクリストにどのようなフィルタリングマスクが格納されるかについて説明するための図である。

【図２８】

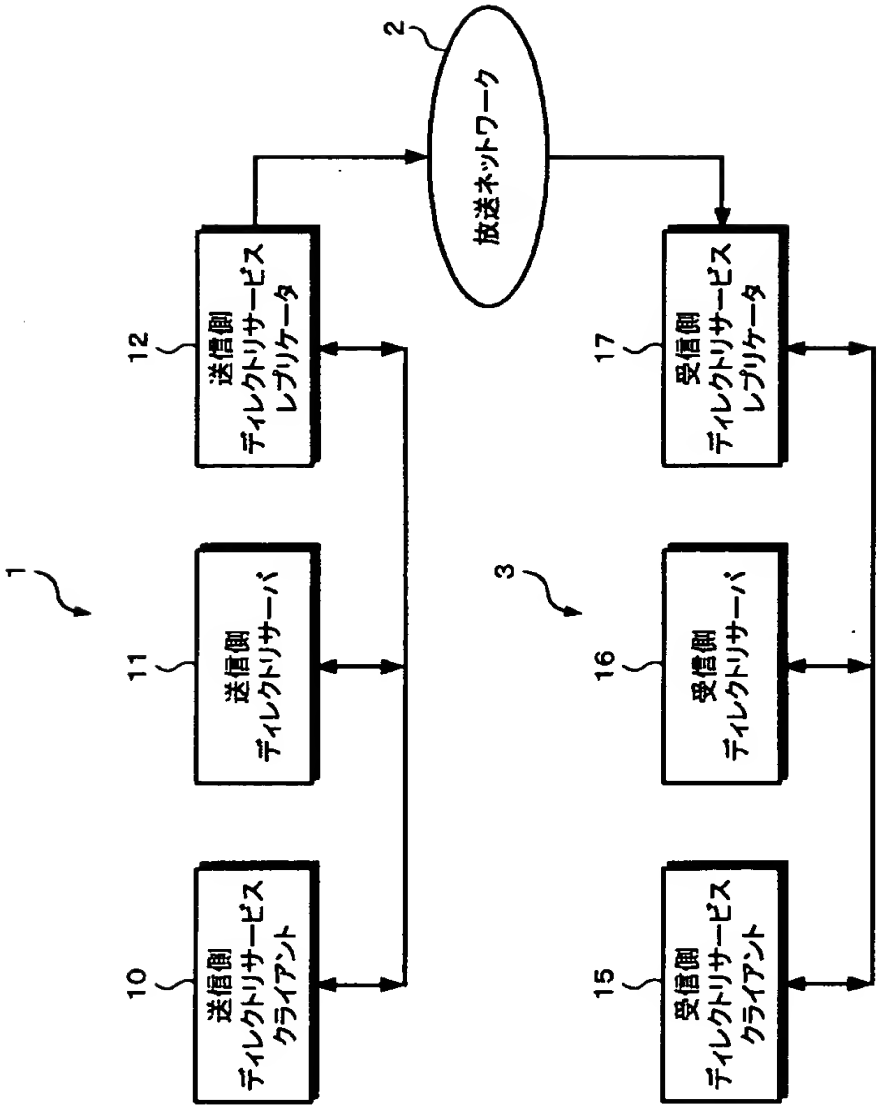
公開鍵証明書の仕組みについて説明するための図である。

【符号の説明】

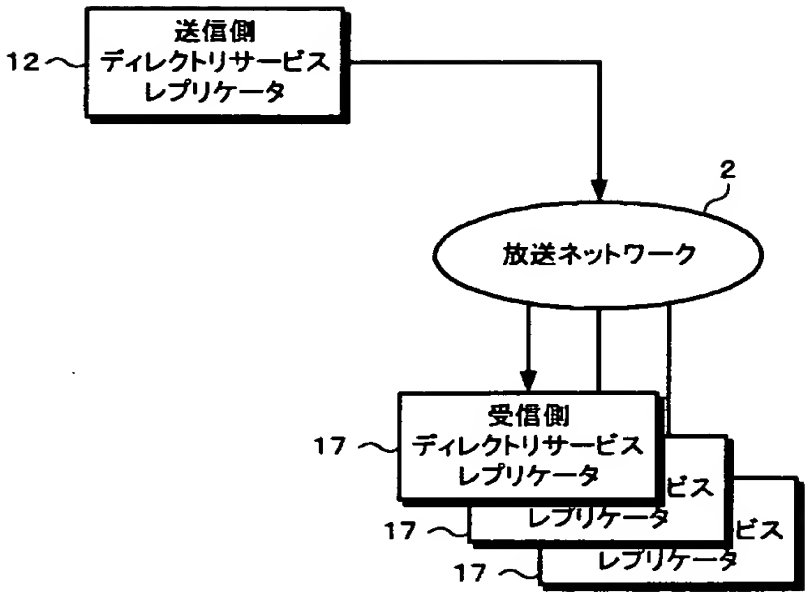
１・・・送信側、２・・・放送ネットワーク、３・・・受信側、１０，１０’・・・送信側ディレクトリサービスクライアント、１１・・・送信側ディレクトリサーバ、１２・・・送信側ディレクトリサーバレプリケータ、１５，１５’・・・受信側ディレクトリサービスクライアント、１６，１６’・・・受信側ディレクトリサーバ、１７，１７’・・・受信側ディレクトリサーバレプリケータ、５０，５０’・・・暗号化通信モジュール、５１・・・通信ネットワーク、ＣＡ＿０，ＣＡ＿１，ＣＡ＿２・・・認証局、ＥＥ＿１，ＥＥ＿２，ＥＥ＿３・・・エンドエンティティ

【書類名】　　図面

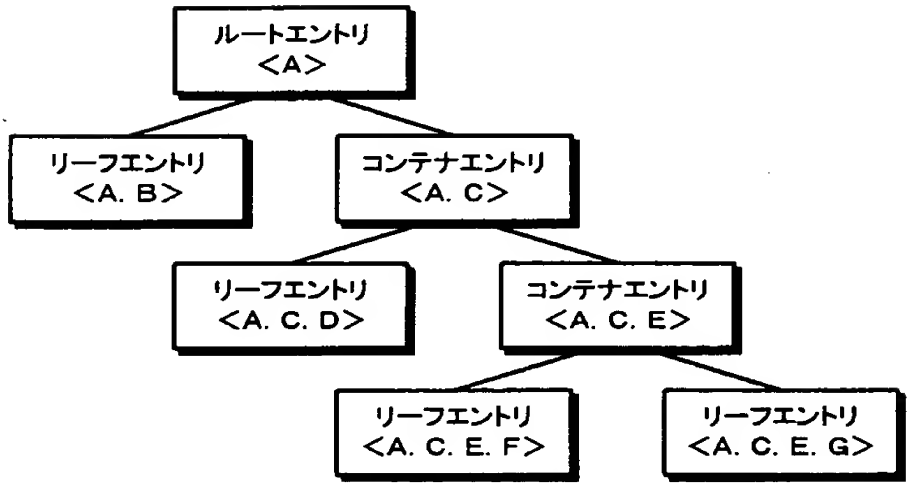
【図１】


2
放送ネットワーク
12
送信側
ディレクトリサービス
レプリケータ
17
受信側
ディレクトリサービス
レプリケータ
1
11
送信側
ディレクトリサーバ
3
16
受信側
ディレクトリサーバ
10
送信側
ディレクトリサービス
クライアント
15
受信側
ディレクトリサービス
クライアント

【図２】

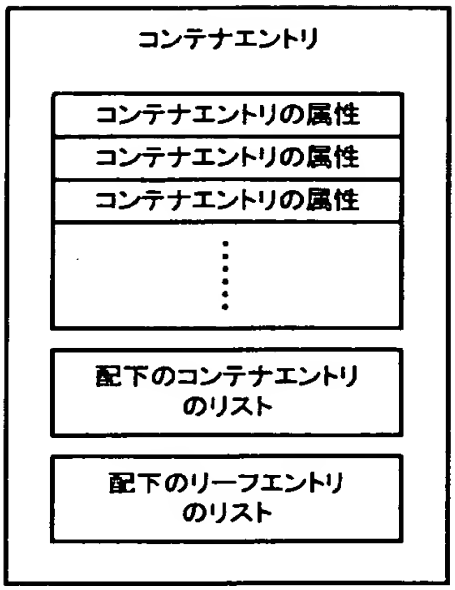

送信側
ディレクトリサービス
レプリケータ
12
2
放送ネットワーク
受信側
ディレクトリサービス
レプリケータ
17
ビス
レプリケータ
17
ビス
レプリケータ
17

【図３】

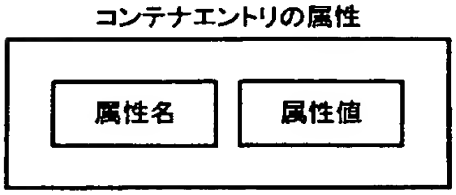

ルートエントリ
<A>
リーフエントリ
<A. B>
コンテナエントリ
<A. C>
リーフエントリ
<A. C. D>
コンテナエントリ
<A. C. E>
リーフエントリ
<A. C. E. F>
リーフエントリ
<A. C. E. G>

【図４】

A

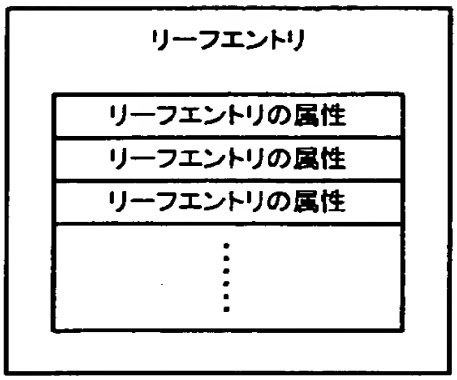

コンテナエントリ
コンテナエントリの属性
コンテナエントリの属性
コンテナエントリの属性
配下のコンテナエントリのリスト
配下のリーフエントリのリスト

B

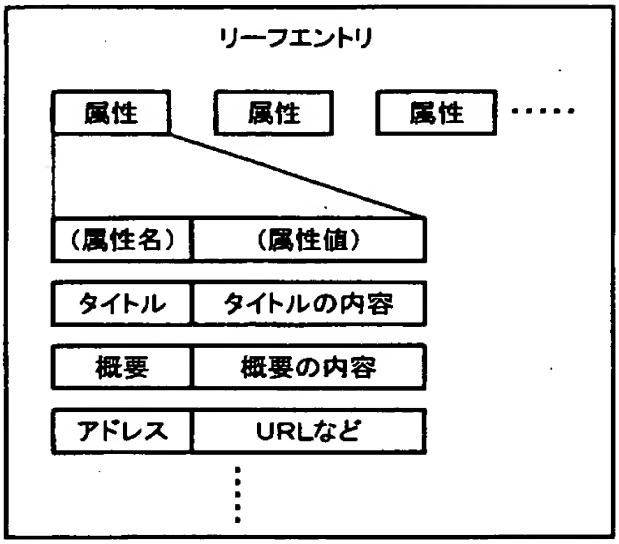

コンテナエントリの属性
属性名
属性値

【図５】

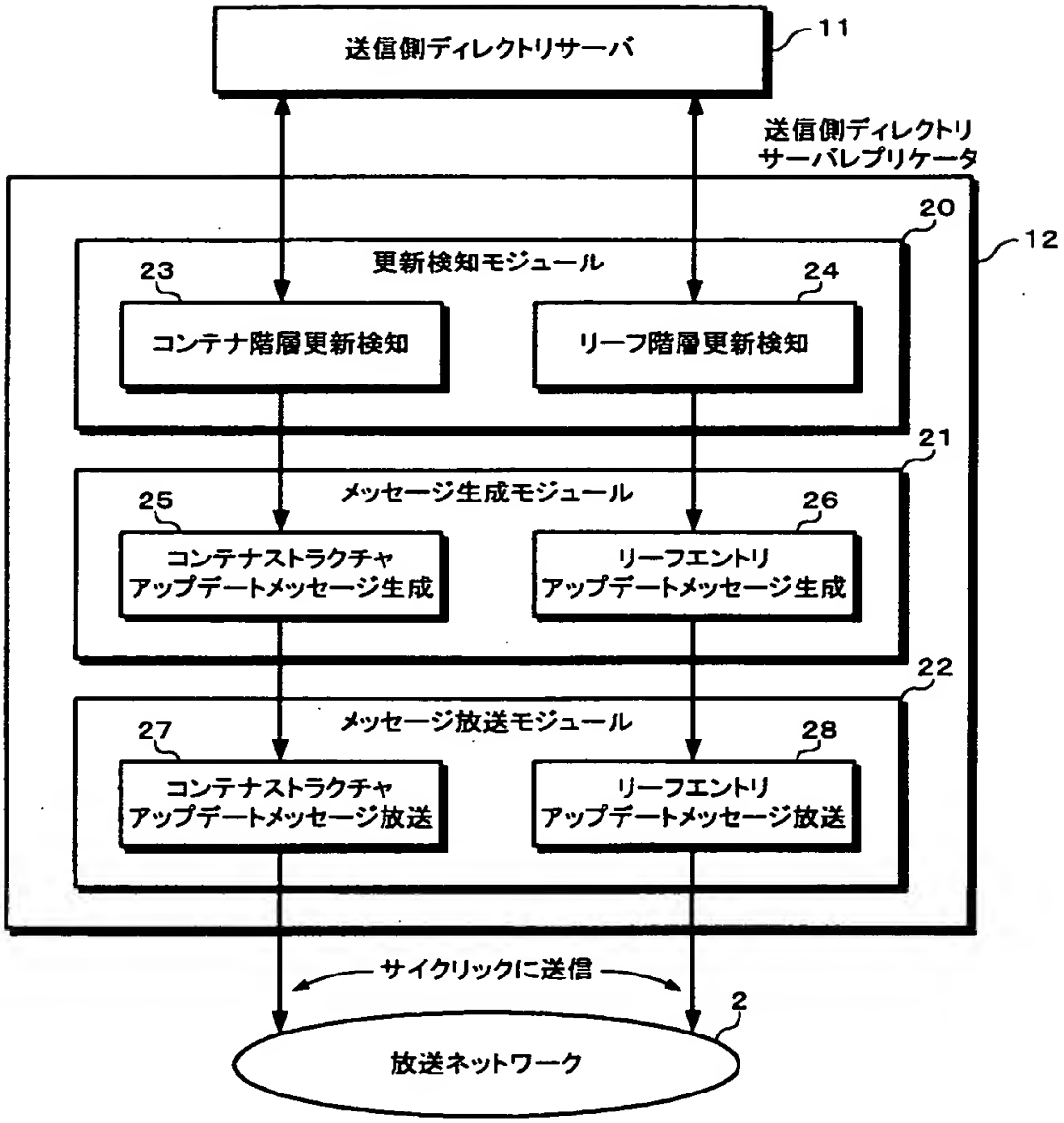
A


リーフエントリ
リーフエントリの属性
リーフエントリの属性
リーフエントリの属性

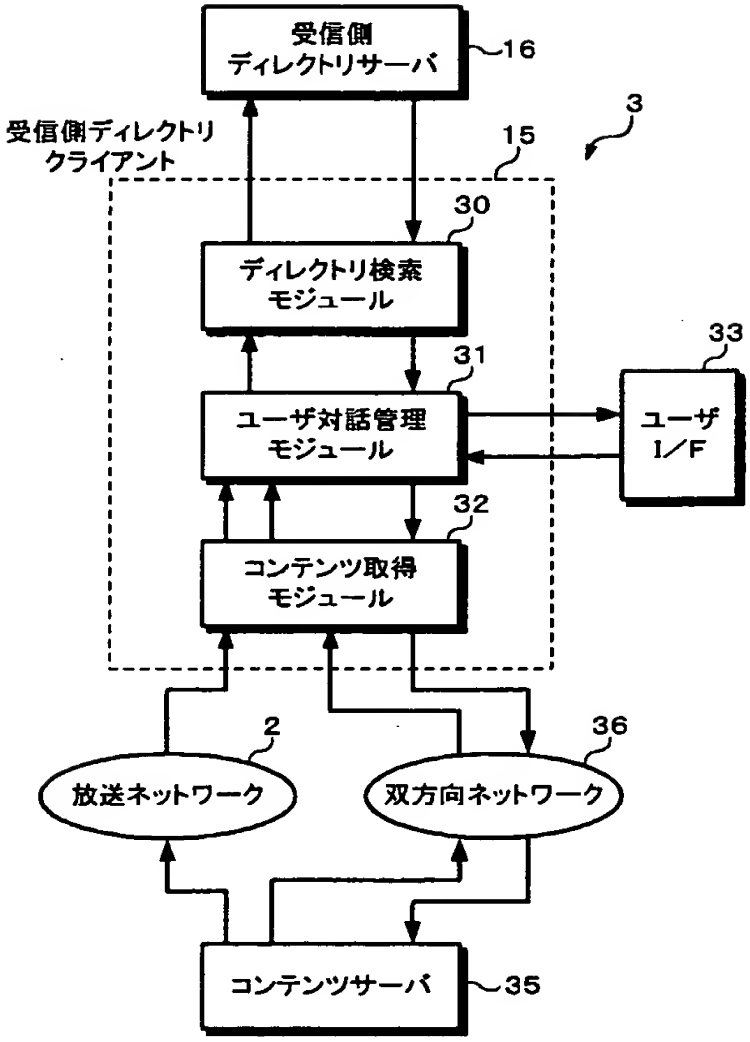
B


リーフエントリ
属性
属性
属性
（属性名）
（属性値）
タイトル
タイトルの内容
概要
概要の内容
アドレス
URLなど

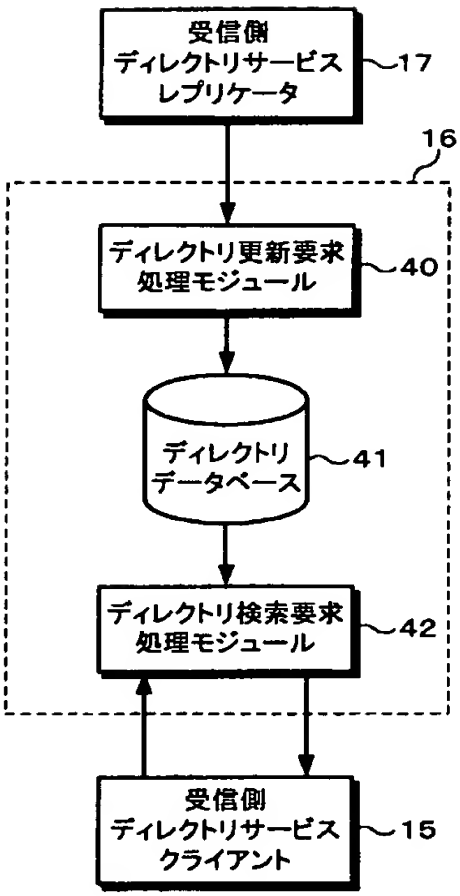
【図６】


送信側ディレクトリサーバ
11
送信側ディレクトリ
サーバレプリケータ
12
20
更新検知モジュール
23
24
コンテナ階層更新検知
リーフ階層更新検知
21
メッセージ生成モジュール
25
26
コンテナストラクチャ
アップデートメッセージ生成
リーフエントリ
アップデートメッセージ生成
22
メッセージ放送モジュール
27
28
コンテナストラクチャ
アップデートメッセージ放送
リーフエントリ
アップデートメッセージ放送
サイクリックに送信
2
放送ネットワーク

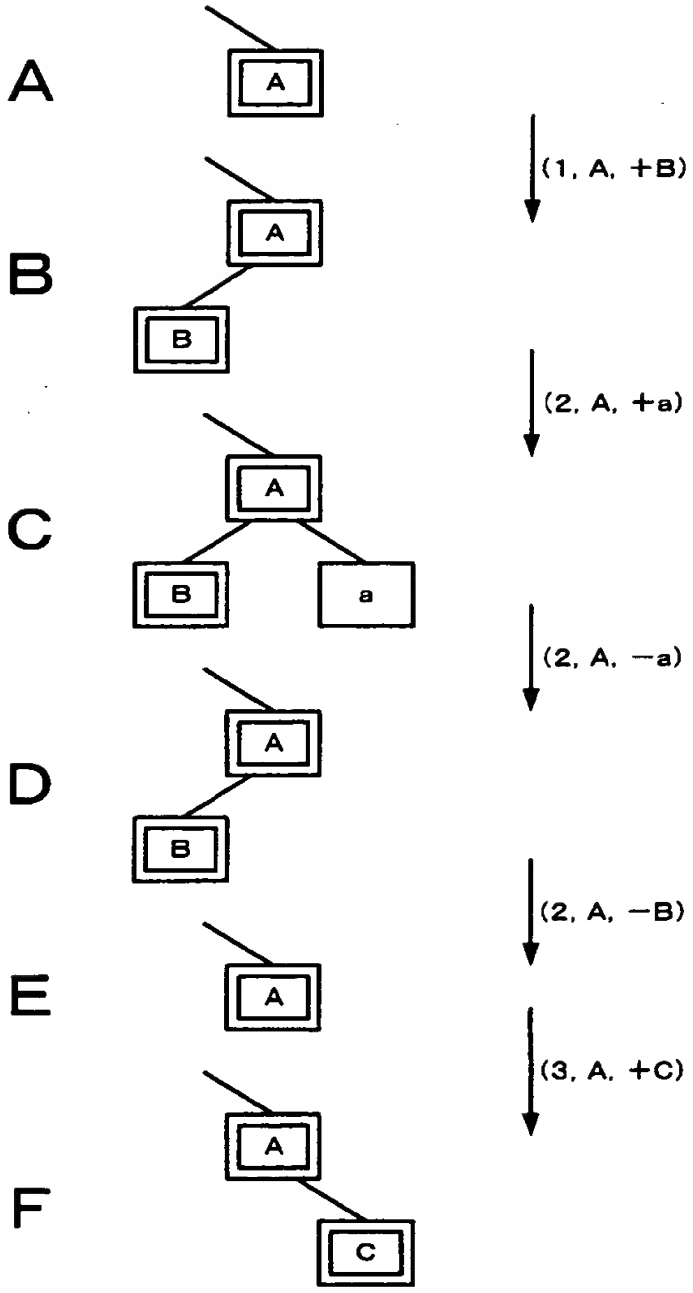
【図７】


受信側
ディレクトリサーバ
16
3
受信側ディレクトリ
クライアント
15
30
ディレクトリ検索
モジュール
33
31
ユーザ対話管理
モジュール
ユーザ
I/F
32
コンテンツ取得
モジュール
2
36
放送ネットワーク
双方向ネットワーク
コンテンツサーバ
35

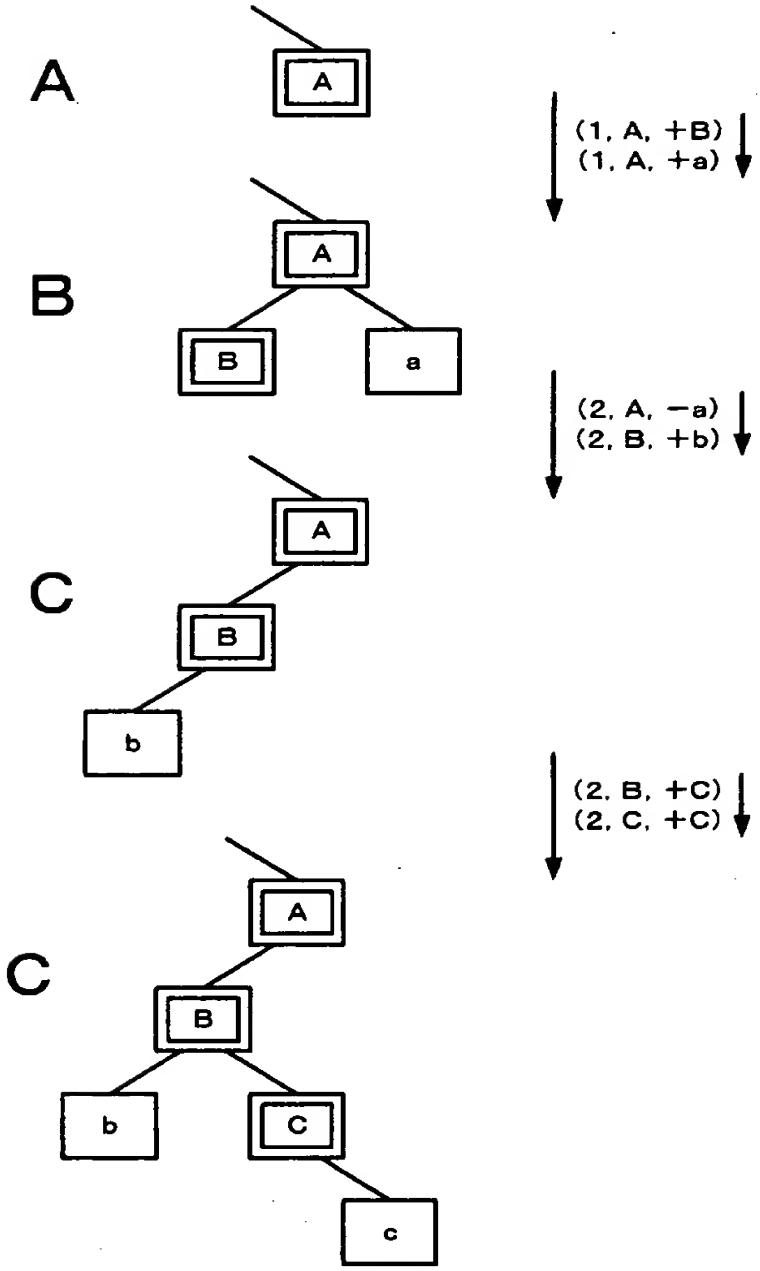
【図８】


受信側
ディレクトリサービス
レプリケータ
17
16
ディレクトリ更新要求
処理モジュール
40
ディレクトリ
データベース
41
ディレクトリ検索要求
処理モジュール
42
受信側
ディレクトリサービス
クライアント
15

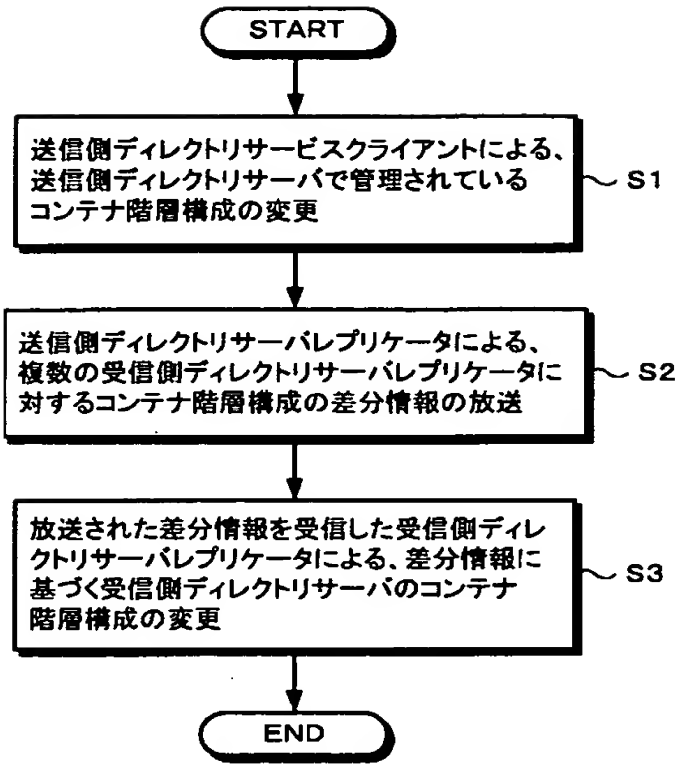
【図９】



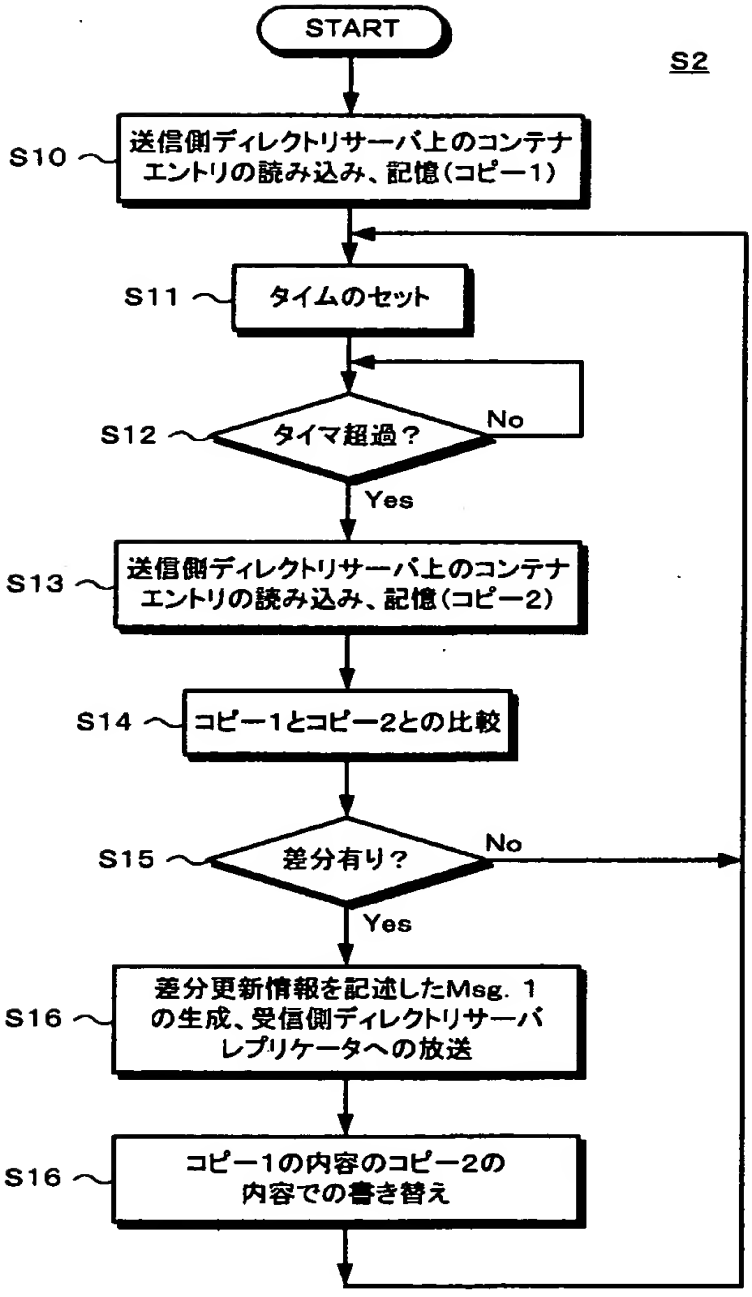
【図１０】



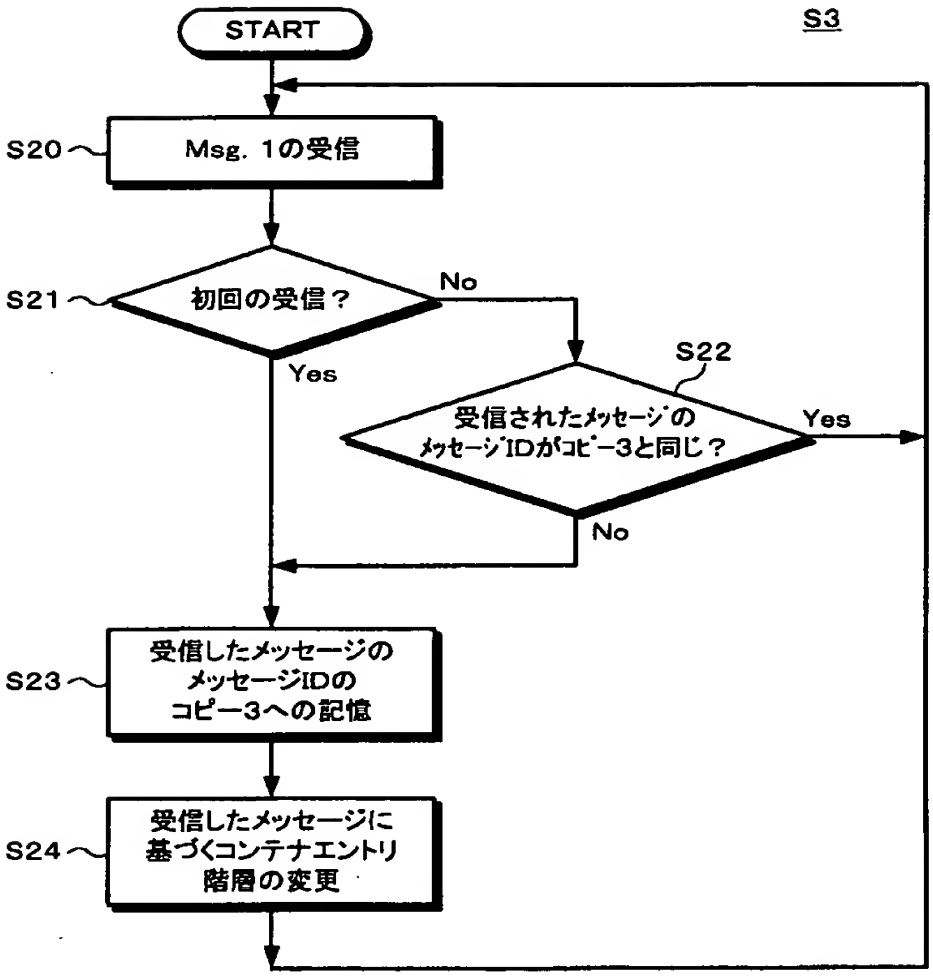
【図１１】


START
送信側ディレクトリサービスクライアントによる、送信側ディレクトリサーバで管理されているコンテナ階層構成の変更
S1
送信側ディレクトリサーバレプリケータによる、複数の受信側ディレクトリサーバレプリケータに対するコンテナ階層構成の差分情報の放送
S2
放送された差分情報を受信した受信側ディレクトリサーバレプリケータによる、差分情報に基づく受信側ディレクトリサーバのコンテナ階層構成の変更
S3
END

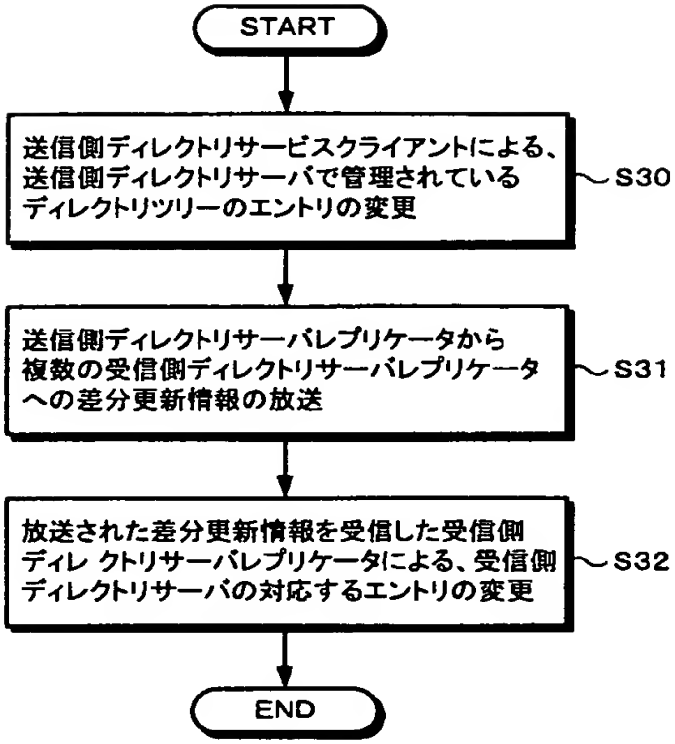
【図１２】


START
S2
S10
送信側ディレクトリサーバ上のコンテナエントリの読み込み、記憶(コピー1)
S11
タイムのセット
S12
タイマ超過？
No
Yes
S13
送信側ディレクトリサーバ上のコンテナエントリの読み込み、記憶(コピー2)
S14
コピー1とコピー2との比較
S15
差分有り？
No
Yes
S16
差分更新情報を記述したMsg. 1の生成、受信側ディレクトリサーバレプリケータへの放送
S16
コピー1の内容のコピー2の内容での書き替え

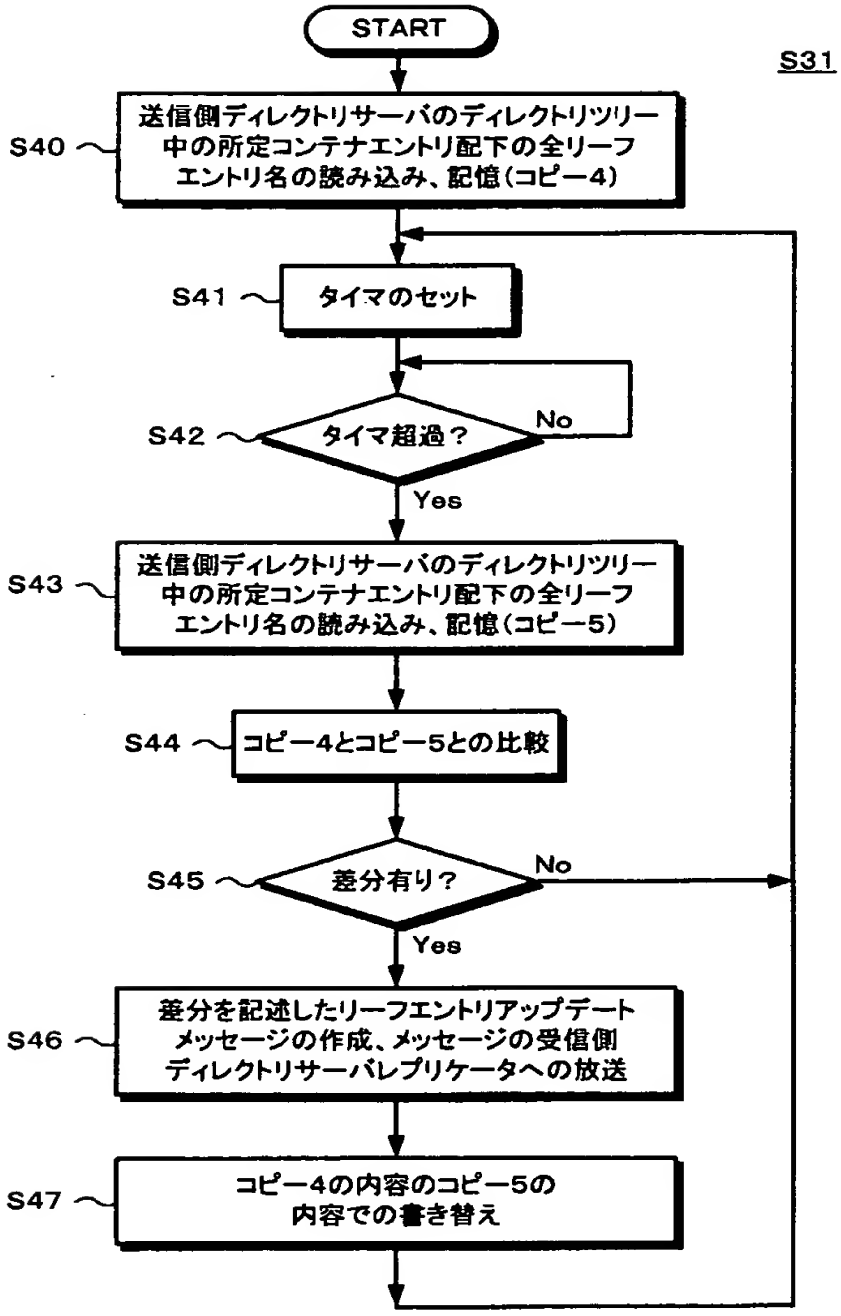
【図１３】


START
S3
S20
Msg. 1の受信
S21
初回の受信？
No
Yes
S22
受信されたメッセージのメッセージIDがコピー3と同じ？
Yes
No
S23
受信したメッセージのメッセージIDのコピー3への記憶
S24
受信したメッセージに基づくコンテナエントリ階層の変更

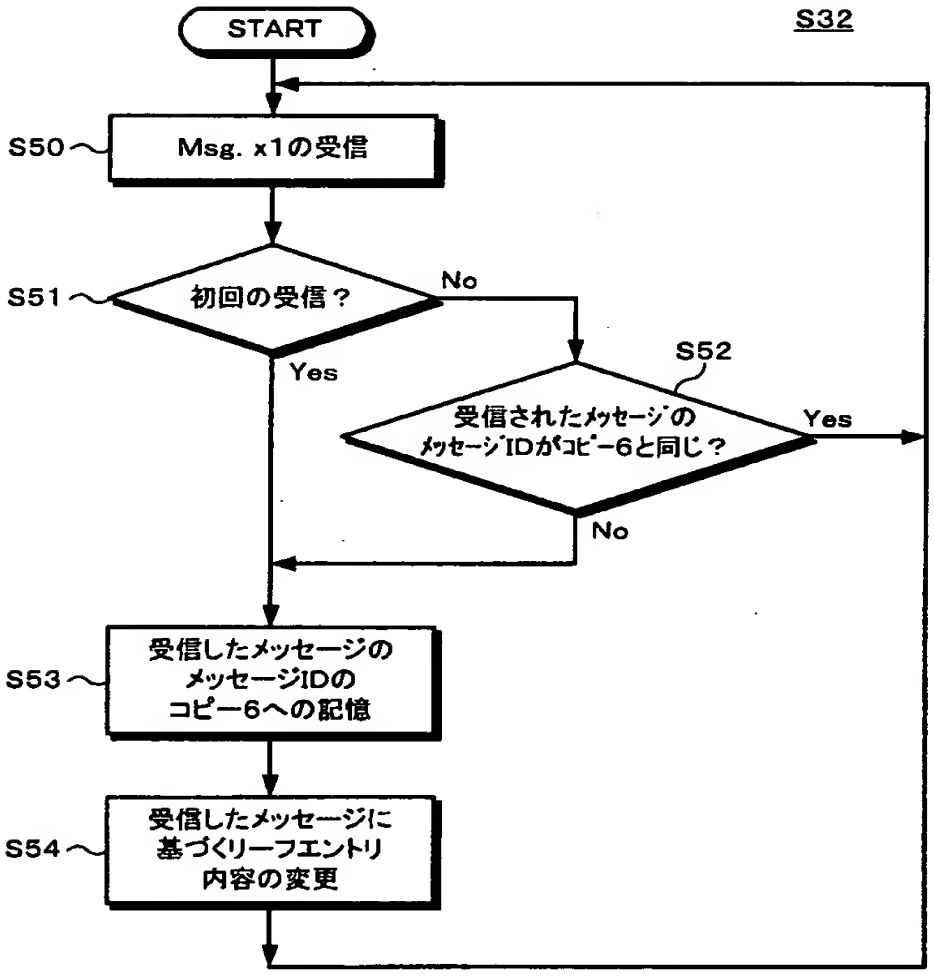
【図１４】


START
送信側ディレクトリサービスクライアントによる、送信側ディレクトリサーバで管理されているディレクトリツリーのエントリの変更
S30
送信側ディレクトリサーバレプリケータから複数の受信側ディレクトリサーバレプリケータへの差分更新情報の放送
S31
放送された差分更新情報を受信した受信側ディレ クトリサーバレプリケータによる、受信側ディレクトリサーバの対応するエントリの変更
S32
END

【図１５】

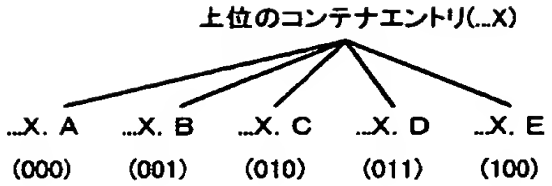

START
S31
S40
送信側ディレクトリサーバのディレクトリツリー中の所定コンテナエントリ配下の全リーフエントリ名の読み込み、記憶（コピー4）
S41
タイマのセット
S42
タイマ超過？
No
Yes
S43
送信側ディレクトリサーバのディレクトリツリー中の所定コンテナエントリ配下の全リーフエントリ名の読み込み、記憶（コピー5）
S44
コピー4とコピー5との比較
S45
差分有り？
No
Yes
S46
差分を記述したリーフエントリアップデートメッセージの作成、メッセージの受信側ディレクトリサーバレプリケータへの放送
S47
コピー4の内容のコピー5の内容での書き替え

【図１６】

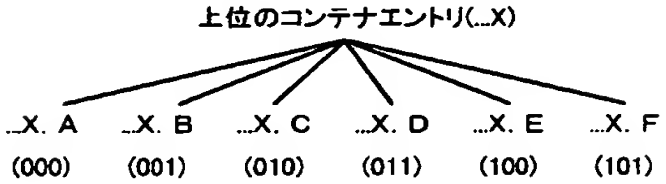

START
S32
S50
Msg. x1の受信
S51
初回の受信？
No
Yes
S52
受信されたﾒｯｾｰｼﾞの
ﾒｯｾｰｼﾞIDがｺﾋﾟｰ6と同じ？
Yes
No
S53
受信したメッセージの
メッセージIDの
コピー6への記憶
S54
受信したメッセージに
基づくリーフエントリ
内容の変更

【図１７】

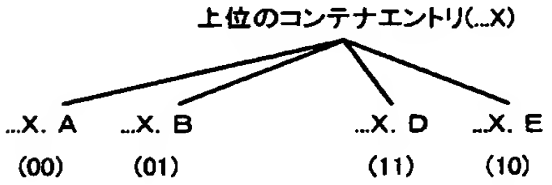
A


上位のコンテナエントリ(...X)
...X. A
(000)
...X. B
(001)
...X. C
(010)
...X. D
(011)
...X. E
(100)

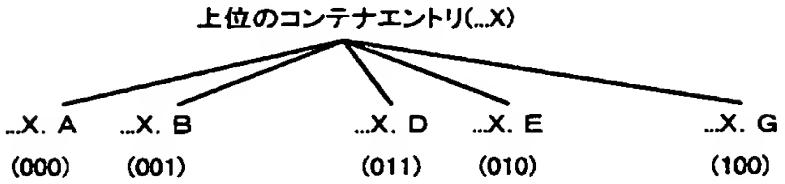
B


上位のコンテナエントリ(...X)
...X. A
(000)
...X. B
(001)
...X. C
(010)
...X. D
(011)
...X. E
(100)
...X. F
(101)

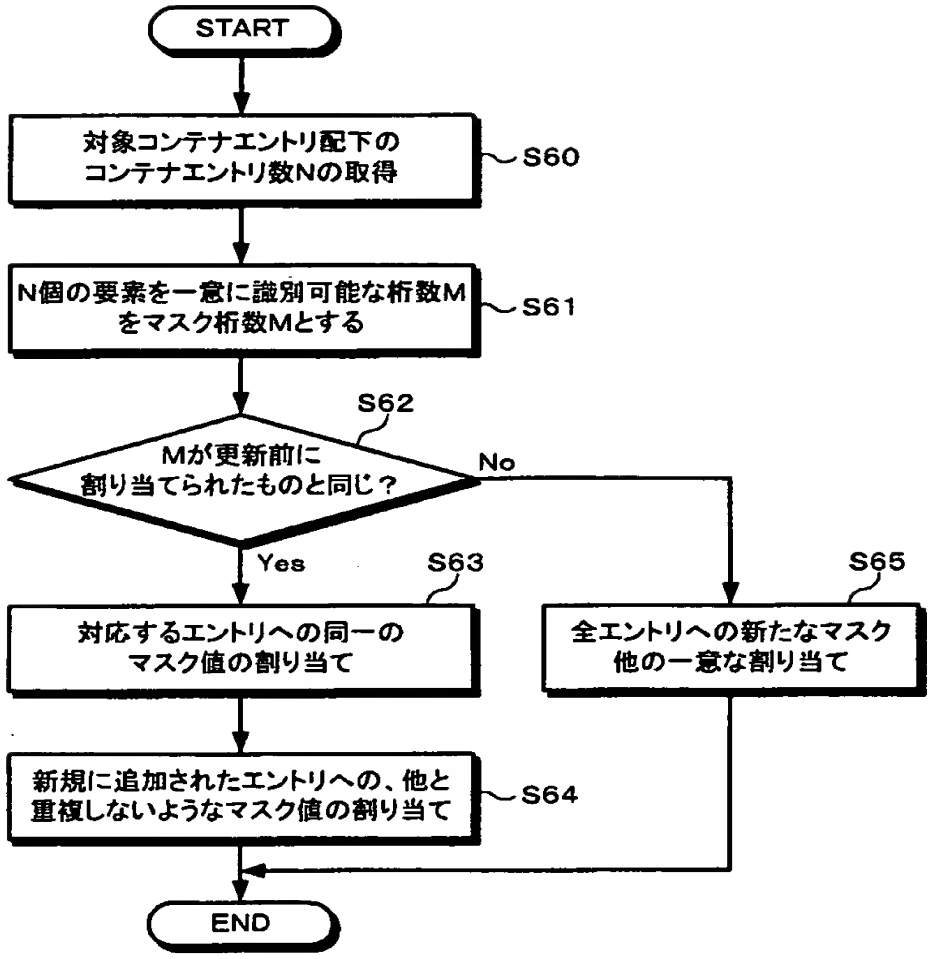
C


上位のコンテナエントリ(...X)
...X. A
(00)
...X. B
(01)
...X. D
(11)
...X. E
(10)

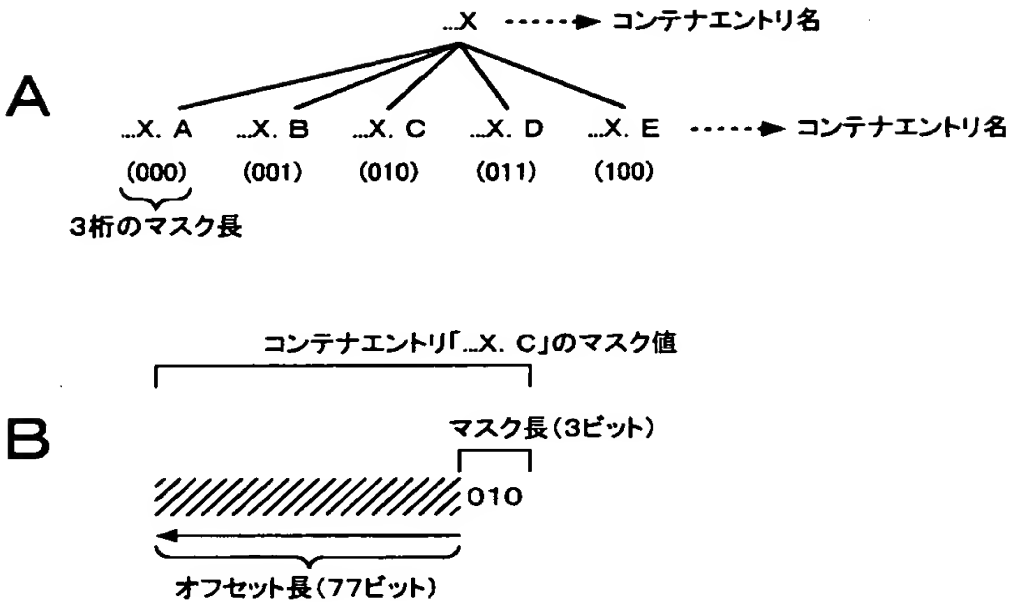
D


上位のコンテナエントリ(...X)
...X. A
(000)
...X. B
(001)
...X. D
(011)
...X. E
(010)
...X. G
(100)

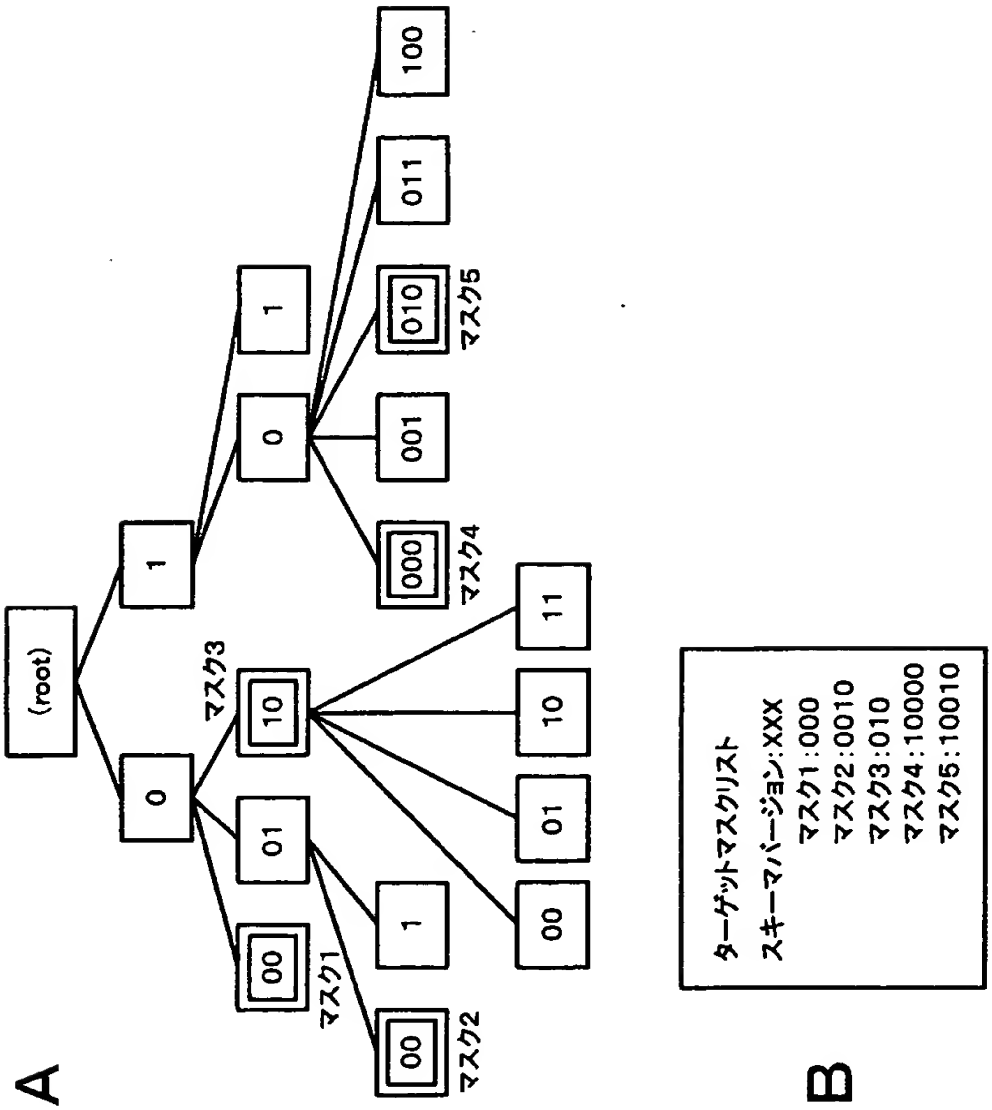
【図１８】


START
対象コンテナエントリ配下の
コンテナエントリ数Nの取得
S60
N個の要素を一意に識別可能な桁数M
をマスク桁数Mとする
S61
S62
Mが更新前に
割り当てられたものと同じ？
No
Yes
S63
対応するエントリへの同一の
マスク値の割り当て
S65
全エントリへの新たなマスク
他の一意な割り当て
新規に追加されたエントリへの、他と
重複しないようなマスク値の割り当て
S64
END

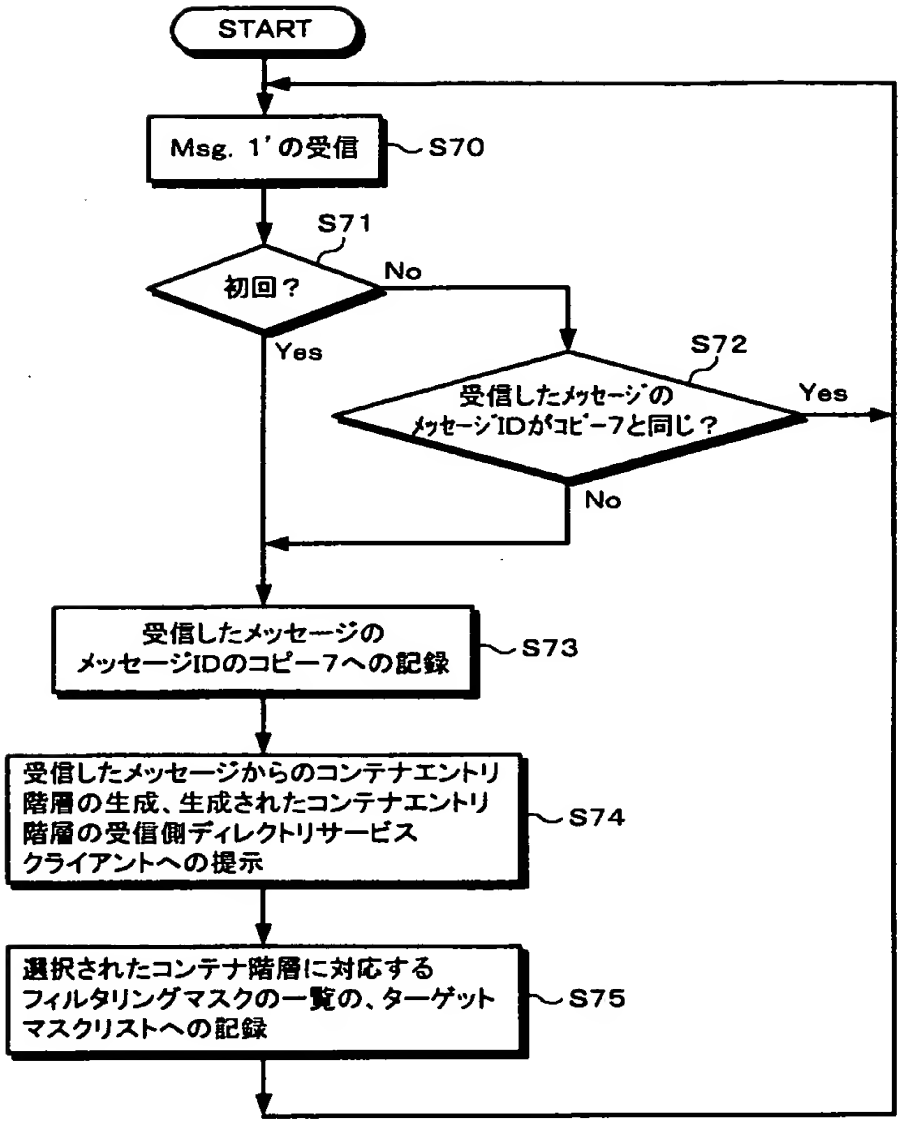
【図１９】


A
...X
コンテナエントリ名
...X. A
...X. B
...X. C
...X. D
...X. E
コンテナエントリ名
(000)
(001)
(010)
(011)
(100)
3桁のマスク長
B
コンテナエントリ「...X. C」のマスク値
マスク長（3ビット）
010
オフセット長（77ビット）

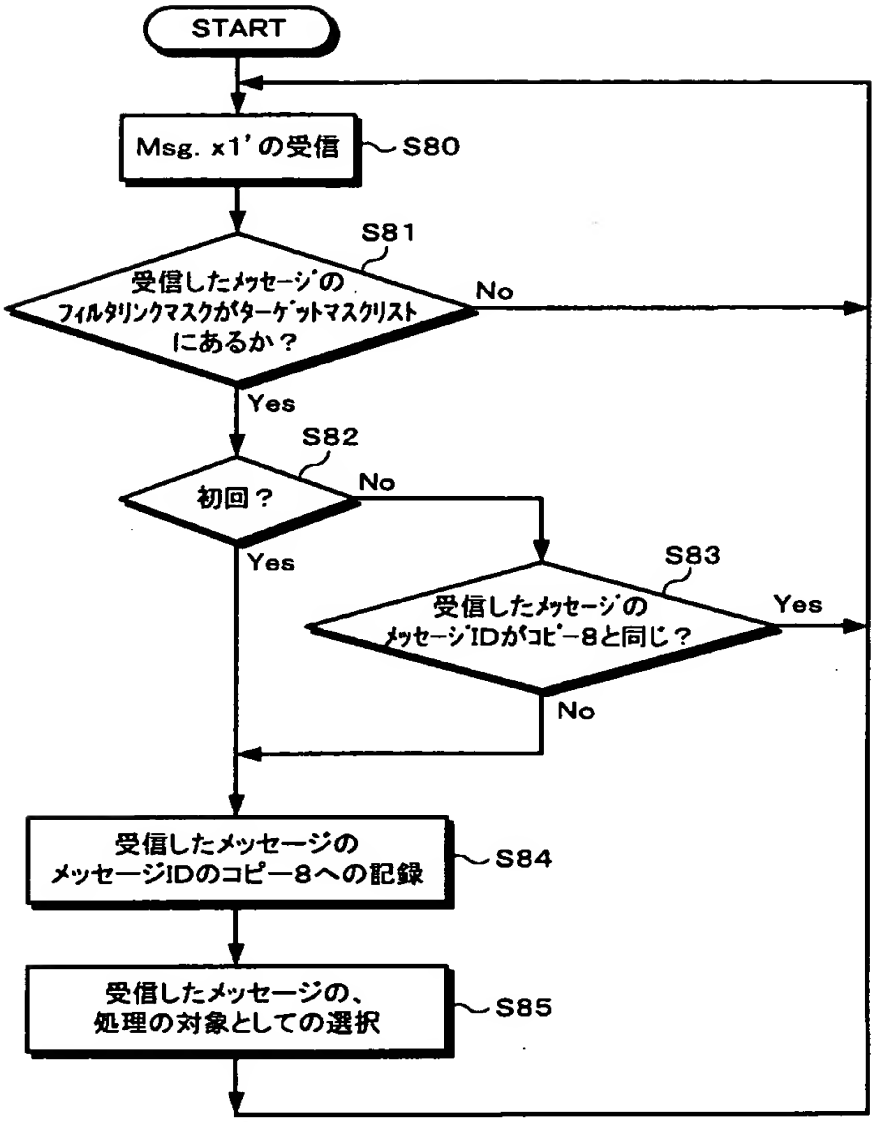
【図２０】


A
(root)
1
0
1
0
100
011
010
マスク5
001
000
マスク4
マスク3
10
01
00
マスク1
1
00
マスク2
11
10
01
00
B
ターゲットマスクリスト
スキーマバージョン:XXX
マスク1:000
マスク2:0010
マスク3:010
マスク4:10000
マスク5:10010

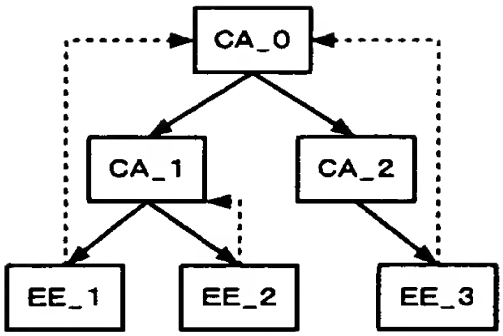
【図２１】


START
Msg. 1'の受信
S70
S71
初回？
No
Yes
S72
受信したメッセージのメッセージIDがコピー7と同じ？
Yes
No
受信したメッセージのメッセージIDのコピー7への記録
S73
受信したメッセージからのコンテナエントリ階層の生成、生成されたコンテナエントリ階層の受信側ディレクトリサービスクライアントへの提示
S74
選択されたコンテナ階層に対応するフィルタリングマスクの一覧の、ターゲットマスクリストへの記録
S75

【図２２】

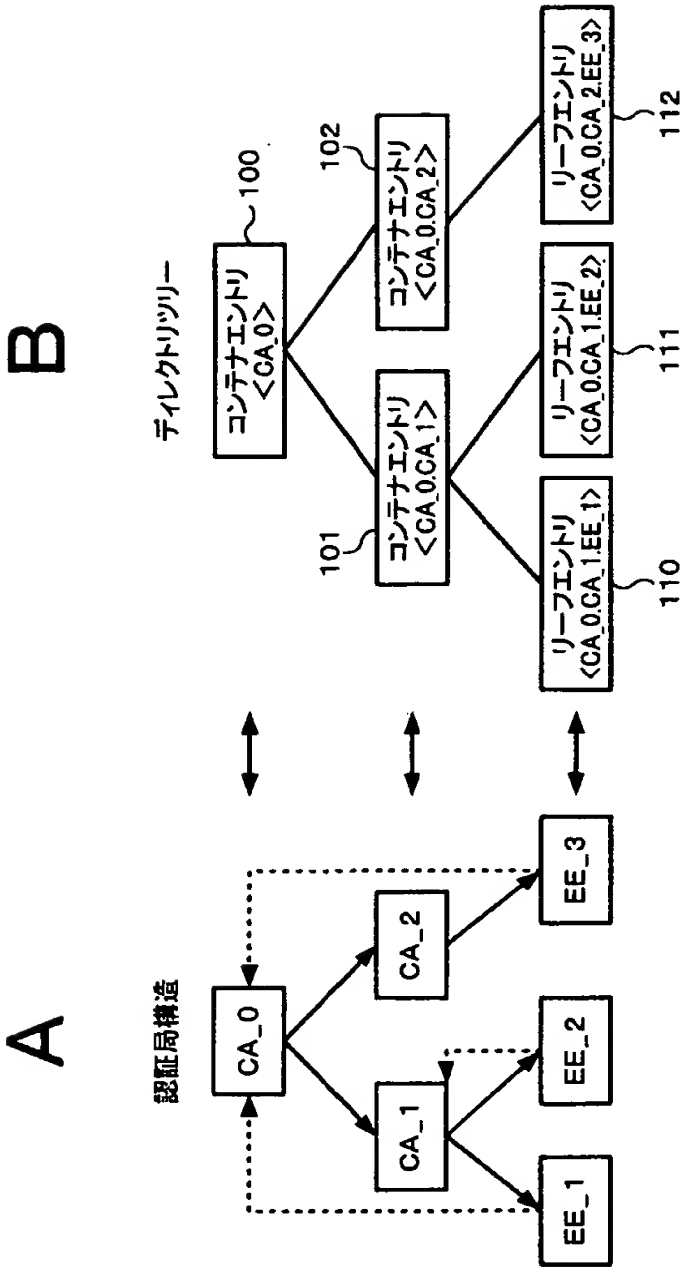

START
Msg. x1' の受信
S80
S81
受信したﾒｯｾｰｼﾞのフィルタリンクマスクがﾀｰｹﾞｯﾄﾏｽｸﾘｽﾄにあるか？
No
Yes
S82
初回？
No
Yes
S83
受信したﾒｯｾｰｼﾞのﾒｯｾｰｼﾞIDがｺﾋﾟｰ8と同じ？
Yes
No
受信したメッセージのメッセージIDのコピー8への記録
S84
受信したメッセージの、処理の対象としての選択
S85

【図２３】

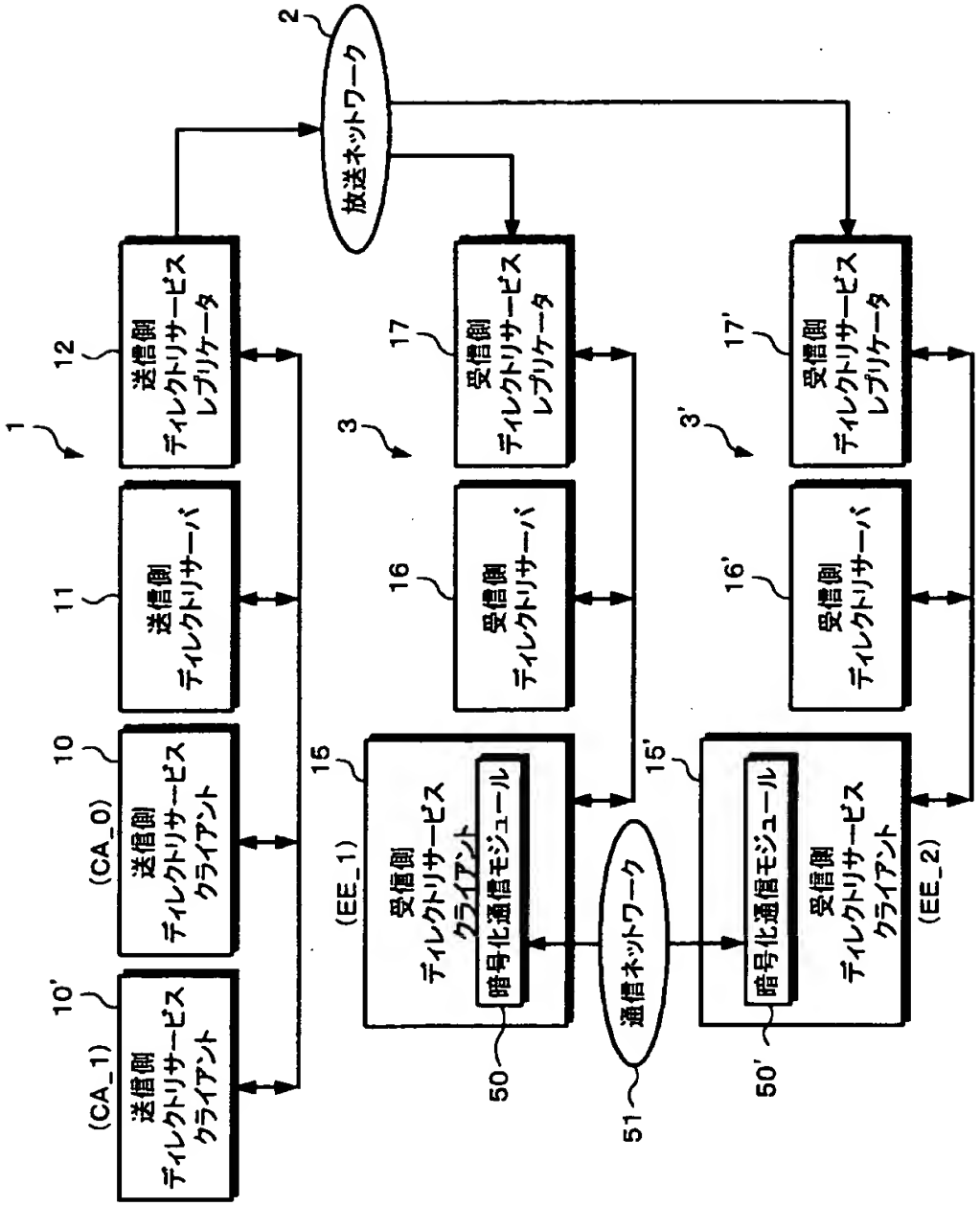


ルート認証局として信頼する：------►
公開鍵証明書を発行する：——►

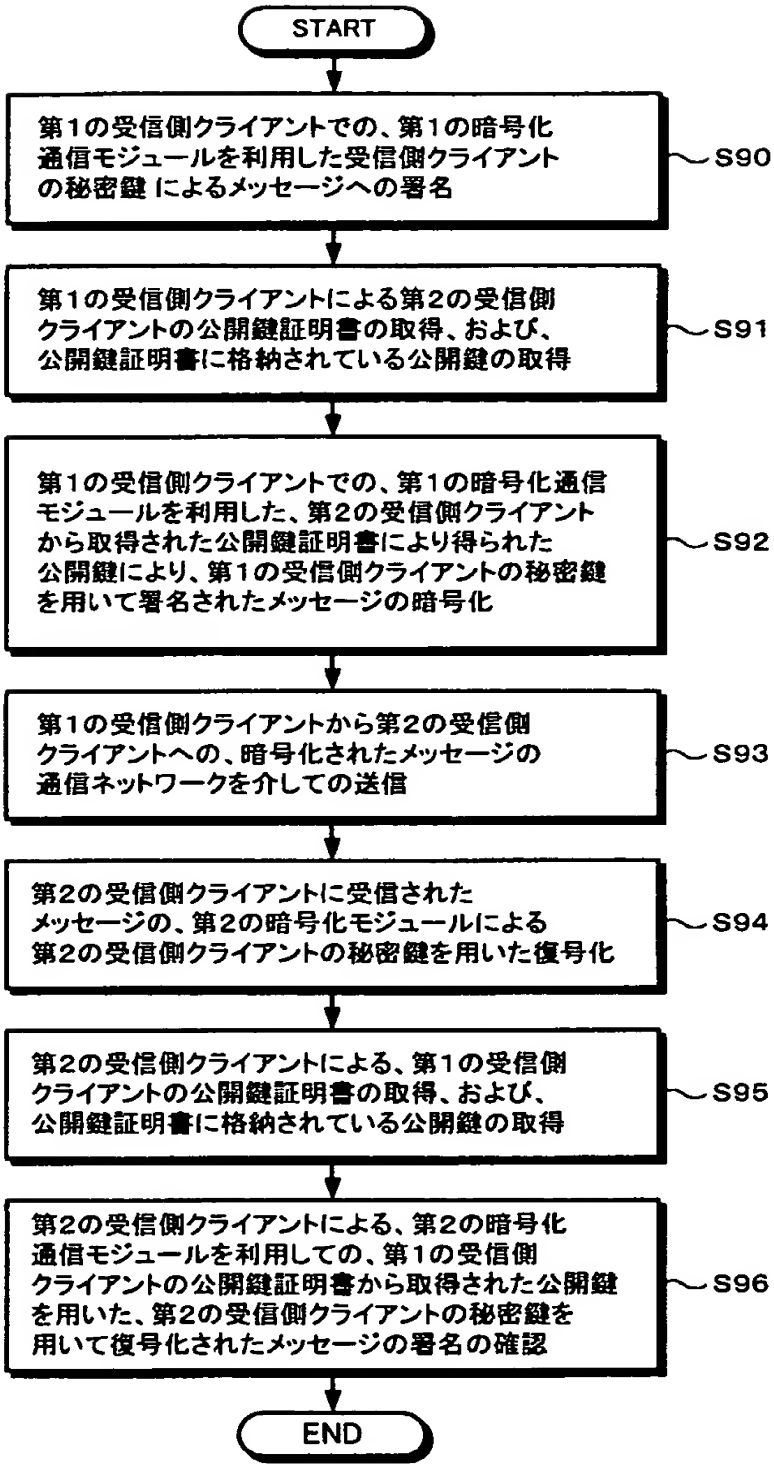
【図２４】


A
認証局構造
CA_0
CA_1
CA_2
EE_1
EE_2
EE_3
B
ディレクトリツリー
コンテナエントリ <CA_0>
100
コンテナエントリ <CA_0.CA_1>
101
コンテナエントリ <CA_0.CA_2>
102
リーフエントリ <CA_0.CA_1.EE_1>
110
リーフエントリ <CA_0.CA_1.EE_2>
111
リーフエントリ <CA_0.CA_2.EE_3>
112

【図２５】

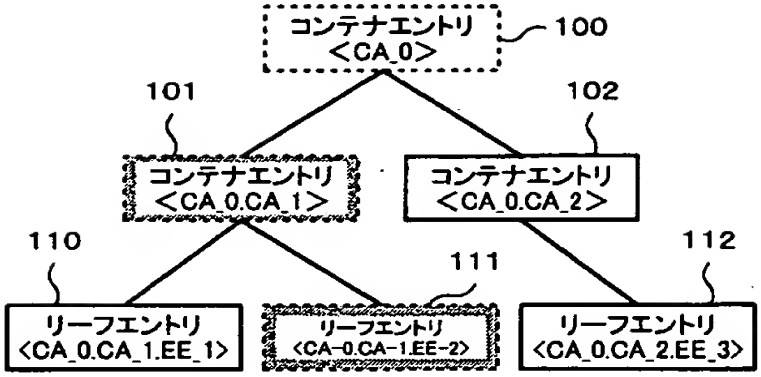

2
放送ネットワーク
1
12
送信側
ディレクトリサービス
レプリケータ
11
送信側
ディレクトリサーバ
10
(CA_0)
送信側
ディレクトリサービス
クライアント
10'
(CA_1)
送信側
ディレクトリサービス
クライアント
3
17
受信側
ディレクトリサービス
レプリケータ
16
受信側
ディレクトリサーバ
15
(EE_1)
受信側
ディレクトリサービス
クライアント
暗号化通信モジュール
50
51
通信ネットワーク
3'
17'
受信側
ディレクトリサービス
レプリケータ
16'
受信側
ディレクトリサーバ
15'
(EE_2)
受信側
ディレクトリサービス
クライアント
暗号化通信モジュール
50'

【図２６】

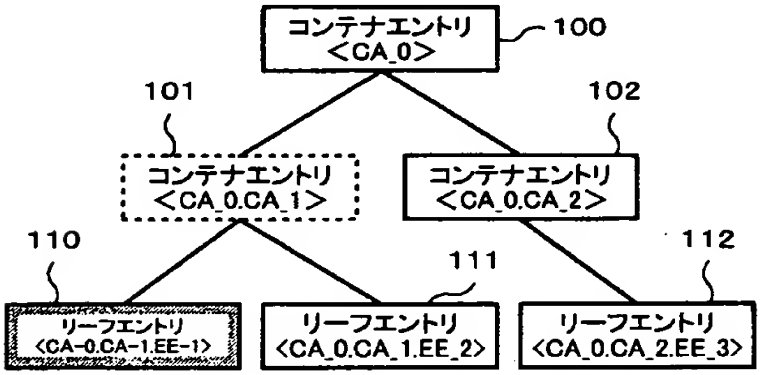

START
第1の受信側クライアントでの、第1の暗号化通信モジュールを利用した受信側クライアントの秘密鍵によるメッセージへの署名
S90
第1の受信側クライアントによる第2の受信側クライアントの公開鍵証明書の取得、および、公開鍵証明書に格納されている公開鍵の取得
S91
第1の受信側クライアントでの、第1の暗号化通信モジュールを利用した、第2の受信側クライアントから取得された公開鍵証明書により得られた公開鍵により、第1の受信側クライアントの秘密鍵を用いて署名されたメッセージの暗号化
S92
第1の受信側クライアントから第2の受信側クライアントへの、暗号化されたメッセージの通信ネットワークを介しての送信
S93
第2の受信側クライアントに受信されたメッセージの、第2の暗号化モジュールによる第2の受信側クライアントの秘密鍵を用いた復号化
S94
第2の受信側クライアントによる、第1の受信側クライアントの公開鍵証明書の取得、および、公開鍵証明書に格納されている公開鍵の取得
S95
第2の受信側クライアントによる、第2の暗号化通信モジュールを利用しての、第1の受信側クライアントの公開鍵証明書から取得された公開鍵を用いた、第2の受信側クライアントの秘密鍵を用いて復号化されたメッセージの署名の確認
S96
END

【図２７】

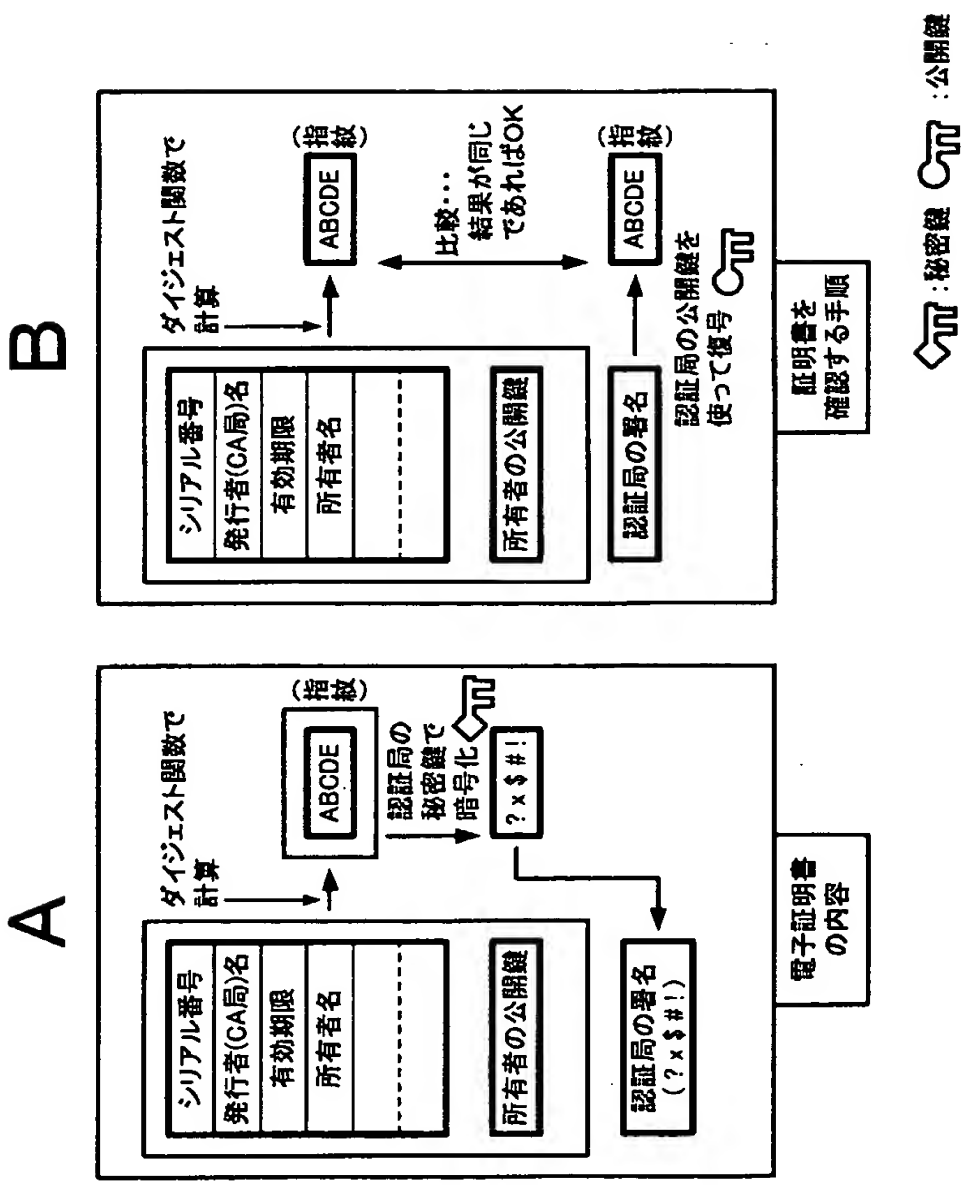
A


コンテナエントリ
<CA_0>
100
101
102
コンテナエントリ
<CA_0.CA_1>
コンテナエントリ
<CA_0.CA_2>
110
111
112
リーフエントリ
<CA_0.CA_1.EE_1>
リーフエントリ
<CA-0.CA-1.EE-2>
リーフエントリ
<CA_0.CA_2.EE_3>

B


コンテナエントリ
<CA_0>
100
101
102
コンテナエントリ
<CA_0.CA_1>
コンテナエントリ
<CA_0.CA_2>
110
111
112
リーフエントリ
<CA-0.CA-1.EE-1>
リーフエントリ
<CA_0.CA_1.EE_2>
リーフエントリ
<CA_0.CA_2.EE_3>

【図２８】


A
ダイジェスト関数で計算
シリアル番号
発行者(CA局)名
有効期限
所有者名
所有者の公開鍵
(指紋)
ABCDE
認証局の秘密鍵で暗号化
?x$#!
認証局の署名(?x$#!)
電子証明書の内容
B
ダイジェスト関数で計算
シリアル番号
発行者(CA局)名
有効期限
所有者名
所有者の公開鍵
(指紋)
ABCDE
比較…結果が同じであればOK
(指紋)
ABCDE
認証局の署名
認証局の公開鍵を使って復号
証明書を確認する手順
:秘密鍵
:公開鍵

【書類名】　要約書

【要約】

【課題】　ＰＫＩディレクトリサーバからＰＫＩディレクトリクライアントへ公開鍵証明書の失効情報の通知を、効率よく行うことができるようにする。

【解決手段】　認証局構造における認証局情報をコンテナエントリに、エンドエンティティ情報をリーフエントリに対応付けて、認証局構造をディレクトリツリーに当てはめる。各エントリの属性として、最新の公開鍵証明書そのもの或いはその取得方法、ならびに、シリアル番号とを持たせる。ある証明書が失効し新しく証明書が発行されたら、新たに発行された証明書とそのシリアル番号とをエントリに格納する。所定の時間経過後に、当該証明書をあるＵＲＬに置くと共に、エントリに格納された証明書をＵＲＬ情報と置き換える。一方、受信側では、必要な証明書を取得するための証明書パスに基づき、フィルタリングマスクを設定する。送信側から送信されたディレクトリ構造は、このフィルタリングマスクにより選択的に受信され、受信側のディレクトリを変更する。送信側からＵＲＬ情報とシリアル番号が格納されたディレクトリツリーが繰り返し送信され、受信側では、フィルタリングマスクで選択されたエントリだけを更新する。最新の公開鍵証明書の参照が分散される。

【選択図】　図２４

出　願　人　履　歴　情　報

識別番号　　〔０００００２１８５〕

１．変更年月日　　１９９０年　８月３０日
［変更理由］　新規登録
住　所　東京都品川区北品川６丁目７番３５号
氏　名　ソニー株式会社